Master Thesis:

Fusion Plasma Thermal Transport
Radial and Poloidal Profile Modeling

Martin Olesen
Study ID number: s051859

PLF – IMM – MAT
DTU, Kgs. Lyngby
June 20, 2011

Technical University of Denmark

Risø DTU
National Laboratory for Sustainable Energy

Plasma Physics and Technology Programme
Building 108/128/129/130, DK-4000 Roskilde, Denmark
Phone +45 46774537
jjra@risoe.dtu.dk
www.risoe.dtu.dk
Supervisor: Volker Naulin

DTU Informatics
Department of Informatics and Mathematical Modelling

Building 321, DK-2800 Kgs. Lyngby, Denmark
Phone +45 45253351, Fax +45 45882673
reception@imm.dtu.dk
www.imm.dtu.dk
Supervisor: Allan Engsig-Karup

DTU Mathematics
Department of Mathematics

Building 303S, DK-2800 Kgs. Lyngby, Denmark
Phone +45 45253031, Fax +45 45881399
MAT-INSTADM@mat.dtu.dk
www.mat.dtu.dk
Supervisor: Anton Evgrafov

Abstract

This thesis was prepared at departments Risø DTU, DTU Informatics and DTU Mathematics at the Technical University of Denmark, in partial fulfillment of the requirements for acquiring the master degree in engineering.

The present work constitutes a numerical study of the Critical Gradient Model (CGM) [21, 9, 22, 15, 24, 23] and the Turbulence Spreading Transport Model (TSTM) [28]. The CGM and TSTM are both heuristic models and are used for a much simplified description of plasma transport by turbulence. In particular, the propagation of thermal perturbations in two distinct types of experiments conducted in the Joint European Torus (shot 55809) are modeled: 1. Modulation of the off-axis localised ion cyclotron resonance heating source. 2. Cold pulse shock induction at the plasma edge via laser ablation. Until recently, no model that incorporates a self-consistent relation between the temperature gradients which drive fluctuations, and the turbulence intensity, has been able to describe both slow heat wave propagation from heat modulation and the fast propagation of a cold pulse, at the same plasma parameters. However, this has been successfully modeled with the TSTM [28].

After establishing a numerical scheme accommodating the special requirements of the CGM and TSTM dynamics, namely efficient handling of stiffness, the chosen scheme is verified. The CGM and TSTM are implemented numerically with Matlab using this scheme, and sought validated by comparing to experiment and results found in the literature [21, 28]. Through radial profile CGM investigations the 1-dimensional (1D) implementation is validated and thereby found fit for extension to include the poloidal cross-section of the modeled fusion plasma. The developed 2D poloidal plane implementation is verified against the 1D implementation. The impact on heat modulation and cold pulse simulation results due to the inclusion of the poloidal dynamics is investigated. A 2D scheme allowing for modeling arbitrary reactor geometries is presented. Reproduction of the TSTM results given in [28] is not achieved.

Acknowledgements

The author would like to thank the supervisors for making possible a master thesis combining three distinct, yet related, facets of a real engineering problem. Thus, within this project

- a challenging sub-problem to a greater and hugely important frontline research effort within fusion physics is treated,
- thorough research of efficient numerical schemes suitable for modeling this sub-problem is conducted,
- while preserving a strictly mathematical angle on the equation conditions in the numerical approximation to the observed dynamics.

Allan Engsig-Karup and Anton Evgrafov have provided good feedback when issues arose during code development, as well as in the countless discussions had regarding the road ahead. During the engineering studies at DTU there have been few opportunities to work with fusion, so a special thanks goes to Volker Naulin for defining a challenging engineering problem within this subject. As with all frontline research, there have been unforeseen challenges along the way. Thus, preparing the present thesis has also been preparation for the challenges one will face working as an engineer. All three supervisors have contributed both advice and encouragement, though never left any doubt about who makes the final decisions. In fact, autonomy has been encouraged from the very beginning of the project, with the bi-weekly meetings serving as an anchor to prevent investigations going off on a tangent.

Notation & Abbreviations

Throughout the thesis, the following notation applies.

- In the equation environment expressions enclosed in brackets, $[\dots]$, should be evaluated prior to taking part in other arithmetic operations, whereas function arguments are enclosed by parenthesis $(\dots)$.
- Vector quantities are denoted in bold type.
- Unit vectors are denoted in bold type with hats.
- In chapter 4, bold type is also used to express corresponding elements in two coupled partial differential equations in a single variable.
- The variable t always denotes time.

Abbreviations that will be introduced where appropriate and thereafter used throughout the thesis include;

CGM	Critical Gradient Model
TSTM	Turbulence Spreading Transport Model
LHS	Left Hand Side
RHS	Right Hand Side
CPU	Central Processing Unit
GPU	Graphics Processing Unit
ODE	Ordinary Differential Equation
PDE	Partial Differential Equation
DFT	Discrete Fourier Transform
BDF	Backward Difference Formula
TR-BDF2	Trapezoidal Rule $2nd$ order accurate Backward Difference Formula
LTE	Local Truncation Error
FDM	Finite Difference Method
FEM	Finite Element Method
FVM	Finite Volume Method
DG-FEM	Discontinuous Galerkin Finite Element Method
MHD	Magneto-hydrodynamics
TEM	Trapped Electron Mode
ITG	Ion Temperature Gradient
L-mode	Low Confinement Mode
NBI	Neutral Beam Injection
ICRH	Ion Cyclotron Resonance Heating
ECRH	Electron Cyclotron Resonance Heating
JET	Joint European Torus
ASDEX	Axially Symmetric Divertor Experiment

Contents

Chapter 1

Theory & Motivation

Within this chapter, the motivation and necessary theory underlying the investigations conducted in chapter 3 are given. In section 1.1 the concept of fusion power is introduced, thus motivating the field of confined fusion research. Section 1.2 outlines the consequences of finite electromagnetic fields on charged particles. This interdependence of electromagnetic field and charged particles is exploited to confine fusion plasma, as explained in section 1.3. The key to making fusion a feasible energy resource, is to confine fusion plasma long enough for sufficent fusion processes to occur. Therefore, it is of the utmost importance to understand the thermal dynamics in the confined plasma. An account of this subject is given in section 1.4, leading to the definitions of two heuristic models describing heat transport in a confined fusion plasma:

- The Critical Gradient Model (CGM), section 1.5.
- The Turbulence Spreading Transport Model (TSTM), section 1.7

The latter model has the advantage over the former of being able to describe both slow and fast thermal transport at the same plasma parameters, as described in section 1.6.

1.1 Fusion Energy

Several sustainable sources of energy have been proposed as bids on how to provide future generations with a clean and stable supply of energy, i.e. with minimal impact on the environment and abundant fuel reserves. Fusion power is one such possibility: By colliding nuclei at speeds high enough to overcome Coulomb repulsion, the nuclei will combine, ejecting any excess nucleons. Provided the resulting nuclear is lighter than nickel, it will weigh less than its nucleons do in free form [27]. In other words; mass energy is released according to Einstein's relation

$$E = mc^2$$

in which energy, E, is postulated to be proportional to mass, m, with the speed of light squared, c^2, as the coefficient of proportionality.

In nuclear physics, the likelihood of interaction between a beam (propagating in the $\hat{\boldsymbol{x}}$ -direction) of particle density n_{beam} incident on a target of particle density n_{target}, is expressed mathematically as the cross-section [7],

$$\begin{aligned}\sigma &= -\frac{\mathrm{d}n_{\mathrm{beam}}}{n_{\mathrm{beam}}}\frac{1}{n_{\mathrm{target}}\mathrm{d}x}\\ &= (\text{collision probability})\ (\text{area per target particle})\\ &= \left[\begin{array}{l}\text{effective area per target}\\ \text{particle for causing collisions}\end{array}\right].\end{aligned} \tag{1.1}$$

The concept outlined by Eq. (1.1) can be extended to the more general case, where both beam and target particles are moving. Denoting the velocities of two particle species as $\mathbf{v}_1$ and $\mathbf{v}_2$, the magnitude of the relative velocity between these is

$$\begin{aligned}v &= |\mathbf{v}|\\ &= |\mathbf{v}_1 - \mathbf{v}_2|.\end{aligned}$$

Particle 1 has mass m_1 and kinetic energy $k_\mathrm{B}T_1$, k_B being Boltzmann's constant and T_1, the temperature of species 1. Likewise, particle 2 has mass m_2 and kinetic energy $k_\mathrm{B}T_2$. Assuming each particle species has Maxwellian distributions characterised by $\beta_1 = \frac{m_1}{2k_\mathrm{B}T_1}$ and $\beta_2 = \frac{m_2}{2k_\mathrm{B}T_2}$, respectively, and defining $\beta \equiv \frac{\beta_1\beta_2}{\beta_1+\beta_2}$, one can write the average value of σv over a Maxwellian distribution characterised by β, as [7]

$$\langle\sigma v\rangle = \left[\frac{\beta}{\pi}\right]^{\frac{3}{2}} \int \mathrm{d}\mathbf{v}\ \exp\left(-\beta v^2\right)\sigma(v)v.$$

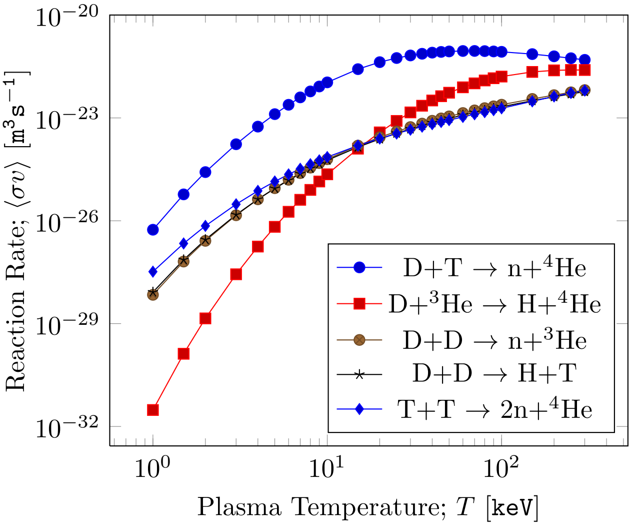


Figure 1.1: Reaction rates for promising fusion fuel candidates. H, D, T, ^{3}He, ^{4}He and n denote hydrogen, deuterium, tritium, helium 3, helium 4 and neutron, respectively. The data is reproduced from [7].

This quantity is called *the reaction rate*, and is a measure of reaction probability between particles with velocities $\mathbf{v}_1$ and $\mathbf{v}_2$. Maxwellian reaction rates are plotted in Fig. 1.1. The data shows that deuterium-tritium fusion reactions have high reaction rate at less heating effort, compared to the other reactions considered. Observe, that there is actually an optimum fusion temperature, beyond which the reaction rate decreases.

Approximately 0.02% of all hydrogen on Earth is deuterium, more than enough to supply the world's population for thousands of years. Tritium, on the other hand, is radioactive with a half-time of 12.3 years and is therefore not found in significant amounts in nature. In the ITER project [14] the fusion reactor will therefore be designed such that tritium is both created and burned inside the reactor [27]. Lithium deposited on the reactor walls will absorb neutrons, thereby becoming unstable, and produce helium and tritium in the process. The combined fuel generation/burn reaction is shown in Fig. 1.2, along with a schematic diagram of the ITER nuclear fusion power plant.

Fig. 1.1 shows, that a temperature around 1 billion Kelvin is required in order for a deuteron-triton reaction to run at the optimum reaction rate. If a deuterium-tritium gas should have any chance to reach such temperatures, the

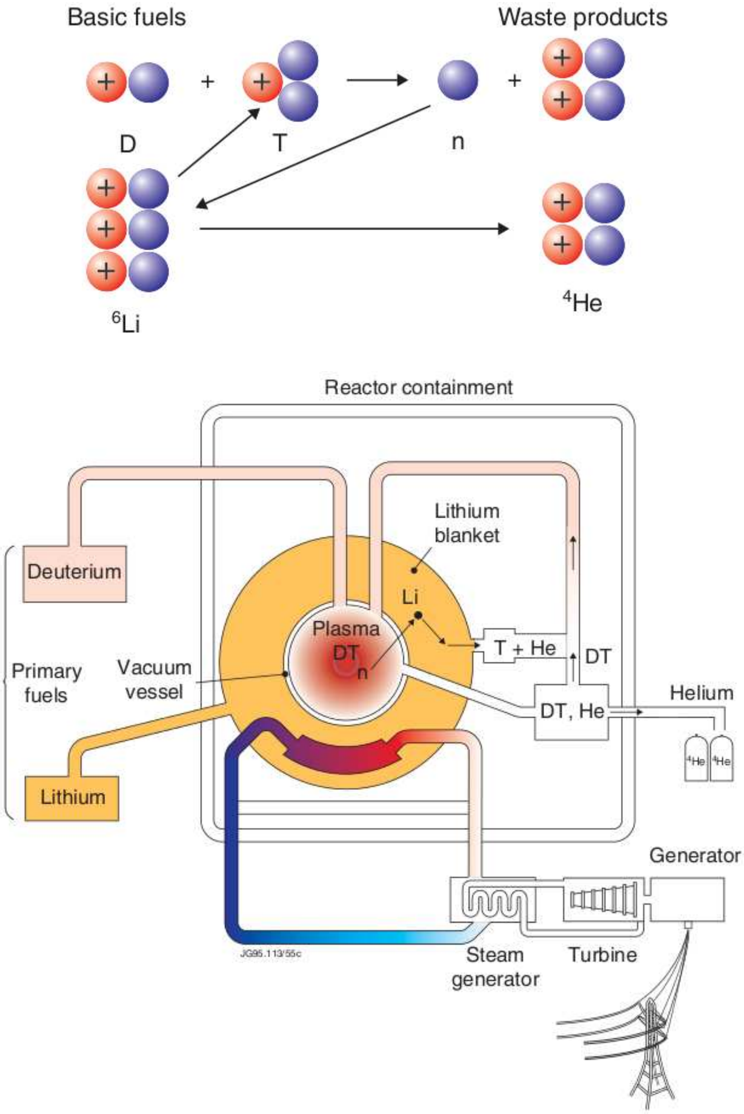


Figure 1.2: Sketch of the deuterium-tritium fusion reaction and the ITER plant [27]. H, D, T, ^{4}He, ^{6}Li and n denote hydrogen, deuterium, tritium, helium 4, lithium and neutron, respectively [27].

gas must be prevented to touch the reactor walls during the heating phase. The particles of the low-pressure gas will transfer the kinetic energy gained in the heating process to the walls upon contact. A clever fusion reactor design that avoids this cooling effect is needed.

The solution is to utilise the fact, that any gas heated sufficiently becomes electrically conducting. When heated gas reaches it's ionisation temperature, a non-negligible number of electrons have become dissociated from nuclei. That is; the gas has undergone a transition from a neutrally charged gas to a soup of ions and electrons, called *plasma*. A plasma can be manipulated using magnetic fields and the whole field of confined fusion physics evolves around utilising this, in order to control a burning fusion process.

1.2 Electromagnetically Induced Plasma Drifts

Force on a point charge due to electromagnetic fields is described by the Lorentz force law;

$$\mathbf{F} = q\mathbf{E} + q\mathbf{v} \times \mathbf{B}, \tag{1.2}$$

where $\mathbf{F} = m\frac{\mathrm{d}\mathbf{v}}{\mathrm{d}t}$ is the Newtonian force, i.e. particle mass, m, times time-derivative of particle velocity, $\mathbf{v}$. The vector fields $\mathbf{E}$ and $\mathbf{B}$ are the electric- and magnetic fields, respectively, experienced by a particle with mass m and charge q. Eq. (1.2) forms the mathematical basis for the description of electromagnetically induced drifts in magnetically confined plasma. To aid the theoretical development of the various drift terms below, Eq. (1.2) is rewritten,

$$\mathbf{F} = \mathbf{F}_{+} + q\mathbf{v} \times \mathbf{B}, \tag{1.3}$$

i.e. the source of force influence $\mathbf{F}_{+}$ besides the $\mathbf{v} \times \mathbf{B}$-term is of arbitrary origin.

For now, the magnetic field is assumed homogeneous in space. If the charged plasma particles in the reactor have non-zero velocity components perpendicular to the magnetic field, $\mathbf{B}$, Eq. (1.3) states that ions and electrons will circle the magnetic field lines in opposite directions, due to their opposing sign charges. The second term in the right hand side (RHS) of Eq. (1.3) accounts for this; particle charge, q, multiplied with the crossproduct between particle velocity vector, $\mathbf{v}$, and local magnetic field, $\mathbf{B}$. The velocity of this *gyro-motion* is rapid, relative to the other drifts discussed below, and is here denoted, $\mathbf{v_g}$.

The arbitrary force, $\mathbf{F}_{+}$ leads to a trivial acceleration in the direction parallel to $\mathbf{B}$. However, in the plane perpendicular to $\mathbf{B}$, the Eq. (1.3) left hand side

(LHS) time-average over one gyro period vanishes;

$$\begin{aligned}\mathbf{F}_\perp &= m\frac{\mathrm{d}\mathbf{v}_\perp}{\mathrm{d}t}\\ &= 0\end{aligned}$$

Remaining LHS and RHS of Eq. (1.3) describing the particle dynamics in a plane perpendicular to **B** are;

$$\begin{aligned}0 &= \mathbf{F}_+ + q\mathbf{v}_\perp \times \mathbf{B}\\ &= \mathbf{F}_+ \times \mathbf{B} + q\big[\mathbf{v}_\perp \times \mathbf{B}\big] \times \mathbf{B}\\ &= \mathbf{F}_+ \times \mathbf{B} - q\big[[\mathbf{B}\cdot\mathbf{B}]\mathbf{v}_\perp - [\mathbf{v}_\perp \cdot \mathbf{B}]\mathbf{B}\big]\\ &\Downarrow\\ \mathbf{v}_\perp &= \frac{\mathbf{F}_+ \times \mathbf{B}}{qB^2} \qquad (1.4)\end{aligned}$$

Suppose $\mathbf{F}_+$ originates from an electric field, i.e. $\mathbf{F}_+ = q\mathbf{E}_\perp$. Inserting in Eq. (1.4) results in the $\mathbf{E} \times \mathbf{B}$ drift,

$$\mathbf{v}_{\mathbf{E}\times\mathbf{B}} = \frac{\mathbf{E}_\perp \times \mathbf{B}}{B^2}. \qquad (1.5)$$

Note that this drift is independent of particle mass and charge.

So far, it has been assumed that **B** is homogeneous in space; plasma particle trajectories are traced out by the drift velocity terms,

$$\mathbf{v} = v_{||}\hat{\mathbf{B}} + \mathbf{v}_\mathbf{g} + \mathbf{v}_{\mathbf{E}\times\mathbf{B}} \qquad (1.6)$$

The RHS terms of Eq. (1.6) represent:

1. Free plasma particle motion along **B**.
2. Fast particle gyration around **B**-field lines.
3. Plasma drift due to perpendicular electric- and magnetic fields.

Suppose $\mathbf{B} = B_z\hat{\mathbf{z}}$ decreases in the $\hat{\mathbf{x}}$-direction. The magnitude of the second RHS term in Eq. (1.3) is equal to the centripetal force for a charged particle, that is

$$|q|v_\perp B_z = \frac{mv_\perp^2}{R_\mathrm{L}}, \qquad (1.7)$$

where R_L is the radius of the circular particle trajectory; the *gyro-* or *Larmor radius*.

Taylor expansion of $\mathbf{B}$, justified when assuming $R_{\mathrm{L}}\frac{\mathrm{d}B_z}{\mathrm{d}x} \ll B_z$, yields

$$\mathbf{B}(\mathbf{r}) = \mathbf{B}(\mathbf{r_0}) + \Big[[\mathbf{r} - \mathbf{r_0}] \cdot \nabla\Big]\mathbf{B} + \cdots . \tag{1.8}$$

Non-zero force contributions are in the (x, y)-plane only. Using Eq. (1.8) with $\mathbf{r_0} = 0$, the force magnitudes are

$$\mathbf{F_x} = q\mathbf{v_y}\left[B_z(0) + x\frac{\mathrm{d}B_z}{\mathrm{d}x}\right] \tag{1.9}$$

$$\mathbf{F_y} = -q\mathbf{v_x}\left[B_z(0) + x\frac{\mathrm{d}B_z}{\mathrm{d}x}\right]. \tag{1.10}$$

Solving Eq. (1.22) for $v_\perp$ yields

$$\begin{aligned} v_\perp &= \frac{|q|B_z R_{\mathrm{L}}}{m} \\ &\Downarrow \\ \nu_{\mathrm{L}} &= \frac{|q|B_z}{2\pi m} && (1.11) \\ &= \frac{\omega_{\mathrm{L}}}{2\pi}, && (1.12) \end{aligned}$$

where ν_{L} is the gyro/Larmor frequency, i.e. ν_{L}^{-1} is the time it takes a plasma particle to revolve one round about a $\mathbf{B}$-field line. Using Eqs. (1.22) and (1.11-1.12), x and y can thus be expressed in terms of sines and cosines according to

$$x = R_{\mathrm{L}}\cos(\omega_{\mathrm{L}}t) \tag{1.13}$$

$$y = R_{\mathrm{L}}\sin(\omega_{\mathrm{L}}t), \tag{1.14}$$

implying that

$$\mathbf{v_x} = v_\perp \sin(\omega_{\mathrm{L}}t)\hat{\mathbf{x}} \tag{1.15}$$

$$\mathbf{v_y} = -v_\perp \cos(\omega_{\mathrm{L}}t)\hat{\mathbf{y}}. \tag{1.16}$$

Inserting (1.13-1.16) in Eqs. (1.9-1.10) yields

$$\mathbf{F_x} = -qv_\perp \cos(\omega_{\mathrm{L}}t)\left[B_z(0) + R_{\mathrm{L}}\cos(\omega_{\mathrm{L}}t)\frac{\mathrm{d}B_z}{\mathrm{d}x}\right]\hat{\mathbf{x}} \tag{1.17}$$

$$\mathbf{F_y} = -qv_\perp \sin(\omega_{\mathrm{L}}t)\left[B_z(0) + R_{\mathrm{L}}\cos(\omega_{\mathrm{L}}t)\frac{\mathrm{d}B_z}{\mathrm{d}x}\right]\hat{\mathbf{y}}. \tag{1.18}$$

Only the guiding centre motion is of interest here, so the force is averaged over one gyroperiod. Since

$$\big\langle \cos(\omega_{\mathrm{L}}t)\big\rangle = \big\langle \sin(\omega_{\mathrm{L}}t)\big\rangle = \big\langle \cos(\omega_{\mathrm{L}}t)\sin(\omega_{\mathrm{L}}t)\big\rangle = 0 \quad \text{and} \quad \big\langle \cos^2(\omega_{\mathrm{L}}t)\big\rangle = \frac{1}{2},$$

the time-averaged forces are

$$\begin{aligned}\langle \mathbf{F_x} \rangle &= -\frac{1}{2} q v_\perp R_\mathrm{L} \frac{\mathrm{d}B_z}{\mathrm{d}x} \hat{\mathbf{x}} \\ \langle \mathbf{F_y} \rangle &= 0.\end{aligned} \tag{1.19}$$

Generalising Eq. (1.19) to arbitrary field orientations is straightforward;

$$\langle \mathbf{F_{arbitrary}} \rangle = -\frac{1}{2} q v_\perp R_\mathrm{L} \nabla B, \tag{1.20}$$

This result shows that charged particles in an inhomogeneous magnetic field are forced down-gradient. Substituting $\mathbf{F}_+$ in Eq. (1.4) with the RHS of Eq. (1.20), and using Eq. (1.22), leads to

$$\begin{aligned}\mathbf{v}_{\nabla B} &= \frac{\mathbf{F}_+ \times \mathbf{B}}{qB^2} \\ &= -\frac{1}{2} v_\perp R_\mathrm{L} \frac{\nabla B \times \mathbf{B}}{B^2} \\ &= \frac{m v_\perp^2}{2|q|} \frac{\mathbf{B} \times \nabla B}{B^3}.\end{aligned} \tag{1.21}$$

This is ∇B drift. Including this drift arising from inhomogenuity in the magnetic field, yields

$$\mathbf{v} = v_{||} \hat{\mathbf{B}} + \mathbf{v_g} + \mathbf{v_{E \times B}} + \mathbf{v}_{\nabla B}.$$

There are additional drifts, such as e.g. curve **B** drift, a kind of centrifugal drift arising from the fact that the magnetic field lines are not straight but curves around in a torus shape. Also, though of vanishing effect, gravitational drift is present. However, the purpose of these derivations is to provide the reader with a flavour of the theory underlying fusion plasma dynamics in a tokamak. See e.g. [7, 27, 4] for a detailed overview of burning fusion plasma dynamics.

The plasma drifts cause asymmetries in the plasma equilibrium which depend on the direction of the magnetic field, **B**, and will need to be kept in check in order to efficiently confine fusion plasma. There are different approaches, and they all involve a complex setup of coils generating magnetic field geometries suitable for confining fusion plasma.

1.3 Tokamak Configuration

One device used for fusion experiments is the tokamak. The reactor chamber is torus-shaped with coils around it. A current is driven through these coils generating a magnetic field in the toroidal direction. If the charged plasma particles in the reactor have non-zero velocity components perpendicular to the toroidal magnetic field, $\mathbf{B_t}$, Lorentz' force law, Eq. (1.2), states that ions and electrons will circle the magnetic field lines in opposite directions, due to their opposing sign charges. Due to the torus shape of the reactor, the magnetic field line density will be greater at the torus centre than at the far edge of the torus. Eq. (1.21) then predicts ∇B drift in the direction perpendicular to the plane spanned by $\mathbf{B_t}$ and ∇B.

In the following, a less mathematical, more intuitive account is given of tokamak confinement.

Compared to that of electrons, the Larmor radius is 40-100 times larger for ions, depending on the ion charge/mass ratio. Solving Eq. (1.7) for R_L results in

$$R_\mathrm{L} = \frac{mv_\perp}{qB_t}, \tag{1.22}$$

i.e. the particle revolution trajectory length $2\pi R_\mathrm{L}$ around $\mathbf{B_t}$ is inversely proportional to B_t. Consider a plasma particle in the tokamak propagating around one magnetic field line. The particle starts its propagation from some point in space between the field line and the torus edge. During the particle's revolution from this point it will feel B_t increase gradually, thereby reducing R_L, until the particle reaches some point in space between that magnetic field line which is now the centre of its revolution, and the torus centre. After this point in space the particle will feel B_t decrease gradually, resulting in increasing R_L. Particles of opposing charges will therefore separate towards the top and bottom of the tokamak, due to the $\mathbf{B_t}$ spatial inhomogenuity. Which charges go where depends on the direction of $\mathbf{B_t}$.

This charge separation results in an approximately vertical electric field $\mathbf{E_v}$, directed from ions to electrons, that – according to Eq. (1.2) – then translates to a force bringing ions and electrons back together, away from the reactor top and bottom walls they were previously heading for. This, however, gives ions and electrons velocity components along the direction of $\mathbf{E_v}$ which the second term in the RHS of Eq. (1.2) then translates to an $\mathbf{E_v} \times \mathbf{B_t}$ drift, towards the outer reactor wall.

To counter this radial ∇B_t drift, a current is driven through the plasma by a large outer coil with high permeability [4]. This current induces a poloidal magnetic field, $\mathbf{B_p}$, that 'squeezes' the plasma and prevents it from touching the reactor walls and cool down, in turn confining the plasma long enough for fusion processes to occur.

Basic tokamak dynamics can be understood by dividing the electromagnetically induced drifts described in section 1.2 into three categories:

1. diamagnetic,
2. $\mathbf{E} \times \mathbf{B}$,
3. ∇B,

of which the diamagnetic terms, i.e. the repulsive plasma response to the externally applied magnetic field, do not contribute to transport. The dominant effect near the divertor plates, where plasma exhaust leaves the confinement region (inside the separatrix; i.e. the boundary between open and closed magnetic field lines), is from the $\mathbf{E} \times \mathbf{B}$ drifts, while the weaker ∇B drifts cause an increase in the magnitude of the radial electric field inside the magnetic separatrix. An important reactor parameter is the safety factor, here denoted by q. This quantity is a measure of the number of times a magnetic field line goes poloidally around the tokamak torus in one toroidal round. The q-profile helps to determine the degree of stability the plasma has against certain instabilities, and is usually near unity in the plasma centre and between 2 and 8 at the edge. The tokamak safety factor can be expressed mathematically as

$$q = \frac{rB_t}{RB_p}, \tag{1.23}$$

where r, R are minor and major radii, respectively, and B_t, B_p the toroidal and poloidal magnetic field magnitudes, respectively. Various heating methods are used, however, the two sources relevant in this work are neutral beam injection (NBI), i.e. a fusion fuel beam shot into the plasma at an angle thus inducing extra toroidal momentum, and electromagnetic radiation at a frequency close to the ion cyclotron resonance (ICRH). Since as much as possible of the heat deposited in the plasma should ignite fusion processes, it is of great interest for researchers to understand the thermal dynamics during tokamak experiments.

1.4 Fusion Plasma Thermal Dynamics

Colliding nuclei generate thermal energy as they merge in the fusion plasma, whereas the plasma looses energy through radiation. A heated plasma exhibits three distinct types of radiation, all of them unavoidable in any fusion reactor. They are

1. Line radiation due to impurities,
2. Cyclotron radiation due to particle Larmor motion,
3. Bremsstrahlung due to Coulomb collisions,

of which 3. usually causes the largest energy loss [8]. The greatest Bremsstrahlung loss occurs during the orbit deflection period while the electron is experiencing accelerated motion due to attraction from the positively charged ion. The fusion process and Bremsstrahlung radiation both scale as the plasma density squared, but the fusion power increases more rapidly for temperatures above 10`keV` [8]. The two processes are important examples of internal sources and sinks in a fusion plasma, which contribute to the overall power balance. Basic diffusive transport of physical quantities such as thermal energy, $\mathcal{E}$, can be described by the *diffusion equation*,

$$\frac{\partial \mathcal{E}}{\partial t} = \nabla \cdot \chi \nabla \mathcal{E} + \mathcal{S}, \tag{1.24}$$

where χ is the *diffusion coefficient* and $\mathcal{S}$ represents heating from external sources. However, many diffusive-like events in nature are not described well by Eq. (1.24), and the thermal dynamics of a fusion plasma is no exception. Whenever parts of a system interfere, nonlinear interactions are involved [35]. In a fusion plasma namely the nonlinear phenomenon of turbulence is believed to play a key role with regard to thermal diffusion (see e.g. [22, 9, 28]), and therefore the RHS of Eq. (1.24), describing the diffusive-driven temporal rate of change of a physical quantity, is modified in the models presented in sections 1.5 & 1.7 in order to account for the nonlinear interactions occuring in fusion experiments. This means that the models cannot be broken down into parts, solved separately and then recombined to get the analytical solution to the full model as dictated by the principle of superposition. A numerical approach must be employed for an efficient analysis of such problems.

A fusion plasma in power balance is characterised by a power balance diffusion coefficient, χ^{PB}. In the ITER project [14], identifying the requirements on density, pressure, temperature and energy confinement of the deuterium-tritium fuel is important in order to produce a favourable overall power balance in the reactor; i.e. to maximise the amount of fusion energy relative to the energy spend on heating the plasma. Low momentum diffusion, high momentum convection and rapid variation of pressure and velocity in space and time is expected to take place, and the challenge is to describe this turbulence in the correct way. To this end, mean field theory of turbulent transport provides a basis for writing reduced transport models for fusion plasmas. With these models, quantities like mean density, velocity and temperature can be calculated. However, reduced transport models sometimes fail and it can be hard to identify the cause of failure. This is why semiheuristic (educated guess) models like the Critical Gradient Model (CGM) have been developed.

1.5 Critical Gradient Model

The basic assumption is, that two different heat transport regimes exist. Recalling Eq. (1.24), this corresponds to having different diffusion coefficients, χ, in each regime. The transition between regimes is assumed to depend on a logarithmic temperature gradient, $\frac{\nabla T}{T}$, threshold; the critical gradient, κ_c. In the subcritical region the diffusion coefficient for a heat pulse is equal to the power balance diffusion coefficient, i.e. $\chi^{\mathrm{HP}} = \chi^{\mathrm{PB}}$, while in the turbulent steep-gradient region $\chi^{\mathrm{HP}} > \chi^{\mathrm{PB}}$ [15]. In other words, neo-classical diffusion (i.e. geometry-dependent diffusion driven by particle collisions) is characterised by the diffusion coefficient χ^{PB}, whereas the increased transport believed to be caused by turbulence developing above κ_c is labeled anomalous transport in the literature [30]. Anomalous transport exceeds neo-classical transport with roughly one to two orders of magnitude [29]. In transport investigations of burning fusion plasma, modulation of the heat sources has proved to be a powerful tool for validation. The transport properties of the plasma are usually assumed to be very close to the transport properties in steady-state for transient events, such as modulation. The CGM is used for interpreting modulation experiments in particular, and the model involves handling of a limited number of free parameters [24, 10]. These free parameters are deduced from experiment and can be compared in different devices. The CGM has been successful in analysing heat modulation when comparing experiment and theory [9].

For simplicity the electron density profile is assumed frozen and flat and the normalised radial coordinate, $\rho = \frac{r}{a}$, is used, r being the physical plasma radius and a being the tokamak minor radius. These conventions apply throughout this work.

The CGM as defined in [28] reads,

$$\frac{\partial T}{\partial t} = \frac{3}{2}\nabla \cdot q^{\frac{3}{2}} T^{\frac{5}{2}} \left[\chi_s \left[\frac{-R\nabla T}{T} - \kappa_c\right] \mathsf{H}\left(\frac{-R\nabla T}{T} - \kappa_c\right) + \chi_0\right] \frac{\nabla T}{T}. \tag{1.25}$$

The transport model is characterised by the stiffness diffusion coefficient, χ_s, the neo-classical (collisional) diffusion coefficient, χ_0, and critical gradient, κ_c, in particular. The remaining parameters are T, the electron temperature, $q = q(\rho)$ is the safety factor described by Eq. (1.23), R is the tokamak major radius, and $\mathsf{H}(\cdots)$ is the Heaviside function. If the critical gradient κ_c is exceeded, the diffusion is increased by the χ_s — the *anomalous flux* — term, thus increasing thermal transport. This tends to prevent build-up of temperature gradients above κ_c, and is the origin of the T-profile stiffness observed experimentally [32]. The χ_0 term represents neo-classical background (collisional) transport, and is in general spatially dependent. All parameters may depend on the electron/ion temperature ratio, the effective charge number and the ratio of the magnetic shear to the safety factor, depending on the underlying instability [9]. Thus, the dependence on various plasma parameters for χ_0, χ_s and κ_c is not explicitly written, since their values are machine dependent and must be identified from relevant experimental data [15]. Furthermore, χ_0 is found to be the quantity with larger uncertainty in its determination; therefore, no speculation on its parametric dependences can be attempted [25]. Recent CGM research reveals that values obtained for χ_0 are usually rather high compared to those predicted by neo-classical transport [22].

To summarise the CGM characteristics, two regimes of heat transport exist;

1. $\chi^{\mathrm{HP}} = \chi^{\mathrm{PB}}$ when $-\frac{R\nabla T}{T} < \kappa_c$,
2. $\chi^{\mathrm{HP}} > \chi^{\mathrm{PB}}$ when $-\frac{R\nabla T}{T} > \kappa_c$.

Thus, the main feature of the CGM is the quasi-equilibrium between the source heating and the turbulence generation: At some point heating will cause the temperature gradient to exceed threshold κ_c and when this happens enhanced turbulent transport kicks in, ‘draining’ temperature towards the plasma edge until the logarithmic temperature gradient is below the threshold and turbulence disappears again. Obviously, the physics of this discontinuously modeled shift between collision-driven and turbulence-driven transport is actually continuous, but the CGM do produce good theoretical [9] and numerical [23] results.

In simulations where a radially dependent source term is added to the RHS of Eq. (1.25), three distinct regions emerge in steady-state:

1. A sub-critical core region, in which there is only neo-classical diffusion,
2. a stiff middle-region, in which the profile slope is close to κ_c,
3. and a super-critical edge region, where the anomalous flux term dominates.

As mentioned, the ranges of variation of the stiffness parameter χ_s and threshold κ_c can be obtained from experiments, and the values used in this work are based on such fits [9].

1.6 Cold Pulse Problem

Perturbation experiments provides an experimental basis for testing theoretical models, thus gaining insight into fusion plasma dynamics. As modulation of the heating sources, a cold pulse applied at the plasma edge by impurity laser ablation is one such experiment. Applying a sudden temperature drop at the plasma edge generates a negative temperature perturbation that propagates into the plasma.

A cold pulse launched from the plasma edge exhibits increased amplitude over about one third of the plasma radius before decreasing, as expected from diffusion. For the Axially Symmetric Divertor Experiment (ASDEX) Upgrade device this puzzling phenomenon is not caused by convection [33]. However, the phase of pulses originating from electron cyclotron resonance heating (ECRH) source modulation and plasma edge cooling in ASDEX Upgrade behaves similarly, and can indeed be explained by the assumption of a CGM [16]. In the Joint European Torus (JET) device the behaviour of cold pulses is quite different from that observed in ASDEX Upgrade. It has been found [28] that the transient propagation of a cold pulse initiated by local cooling, near the plasma edge, produced by either impurity laser ablation or deuterium shallow pellet injection [23], is much faster than the propagation of the heat modulation wave under same plasma conditions. In spite of the successful application of local models such as the CGM to theoretically reproduce the experimentally observed slow propagation of heat waves originating from modulation of the ICRH source, the model fails to account for both slow heat modulation wave propagation and fast propagation of a cold pulse, at the same set of plasma parameters. These two experiments have been conducted at JET. The heat modulation and cold pulse experiments were done simultaneously in the same discharges, to minimise the sources of errors [10, 21]. The cold pulses are observed to propagate very fast from plasma edge to centre even though the temperature gradient is below threshold, κ_c, in a large region of the core, while heat modulation waves are observed to slow down in this region [21]. JET cold pulses only slow down when encountering a region of negative magnetic shear, i.e. a region with reversed safety factor profile, is present in the core [23].

Thus, fusion plasma can exhibit both slow quasidiffusive and fast ballistic-like behaviour, posing quite a challenge to theory.

Several attempts have been proposed to improve reliability of transport predictions, a discussion of which is given in [10]. The highlights are repeated in this section in order to stress the qualities of the model presented in section 1.7. Mixing-length and quasilinear theories are justified by closure schemes that rests on the assumption of Gaussian statistics. However, Gaussian statistics contradicts intermittency, i.e. the alternation of phases of apparantly chaotic dynamics. The turbulence fluid regime is characterised by chaotic, stochastic property changes; low momentum diffusion, high momentum convection and rapid variation of pressure and velocity. Thus, intermittency is the main argument against developing a mean field theory based on a combination of mixing-length and quasilinear theories to solve the cold pulse problem.

Fractional diffusion based transport models have been suggested [5], and these nonlocal models are indeed able to exhibit both diffusive and ballistic behaviour. However, fractional kinetics models are usually considered in the test-particle limit, meaning that a given statistics of fluctuations is assumed. Moreover, it remains unclear which physical process is responsible for the nonlocality. It would be better to have a model that incorporates a self-consistent relation between the temperature gradients which drive fluctuations, and the turbulence intensity; a set of partial differential equations (PDEs) that describe the whole dynamics, both slow propagation from heat modulation and fast propagation of a cold pulse [10].

The CGM can be modified to exhibit such bivalent character by coupling the temperature transport equation to an evolution equation describing the turbulence intensity. Assuming that the turbulent energy propagates on a much faster time-scale, one may be able to account for the fast cold pulse response, thus circumventing the issue of unphysical nonlocality in the temperature transport equation. One model that takes this approach in order to theoretically reproduce the experimentally observed heat modulation and cold pulse fusion plasma response, is the Turbulence Spreading Transport Model (TSTM). This model is presented and motivated in the next section.

1.7 Turbulence Spreading Transport Model

At present, no physics based explanation has been given of the fast cold pulse propagation observed in JET. A cold pulse produces locally a strong temperature gradient, and thereby, as dictated by the CGM, Eq. (1.25), a strong local increase of transport. In addition, cold pulses also create a drop of the electron/ion temperature ratio [24] which can affect ion transport through the ion temperature gradient (ITG) mode stability and influence cold pulse propagation by the coupling between ion and electron channels [17].

It is believed [24], that ECRH modulation yields the most reliable results on electron heat transport. Though, experiments utilising ICRH heating in the mode conversion scheme for cold pulse studies – such as those carried out at

JET – indicate that commonly used models are probably not describing the transport completely. The TSTM is a heuristic model, including nonlocality, that aims at improving the description of thermal transport.

Nonlocality is not physical; a certain point in the plasma cannot instantaneously be affected by plasma conditions at a point at the other end of the plasma torus. However, if nonlocality is interpreted as interference happening at much shorter time-scales compared to time-scales of local interference, a model implementing this could be physically sound. And the TSTM presented in detail below do indeed seem to account for both heat wave (slow) and cold pulse (fast) thermal pulse propagation speeds [28].

To accomodate the physics of heat wave and cold pulse propagation in a fusion plasma, nonlocality is introduced via turbulence spreading. Rather than using the growth rate of the underlying instability, γ,

$$\gamma\left(T, \nabla T\right) = \lambda\sqrt{\frac{-R\nabla T}{T} - \kappa_c} \qquad \text{for} \qquad \frac{-R\nabla T}{T} > \kappa_c, \tag{1.26}$$

$$\gamma\left(T, \nabla T\right) = -\frac{\lambda}{10}\sqrt{\frac{R\nabla T}{T} + \kappa_c} \qquad \text{for} \qquad \frac{-R\nabla T}{T} < \kappa_c, \tag{1.27}$$

to determine the fluxes of transported quantities (as in the CGM), the local intensity of the turbulent fluctuations drive these fluxes in the TSTM [28]:

$$\frac{\partial E}{\partial t} = \nabla \cdot D_0 E \nabla E + \gamma E - [\gamma_0 + \beta E]E, \tag{1.28}$$

$$\frac{\partial T}{\partial t} = \nabla \cdot \left[-CET\tanh\left(\gamma\right) + \frac{3}{2}\chi_0 T^{\frac{5}{2}} q^{\frac{3}{2}}\frac{\nabla T}{T}\right]. \tag{1.29}$$

In the first term in Eq. (1.28), the diffusion of turbulent energy, E, is taken to be nonlinear and proportional to the constant D_0, second term reflects that the turbulent energy input rate is taken to be proportional to the logarithmic temperature gradient dependent growth rate γ, and the third term constitutes weak damping of the turbulent energy (γ_0) and nonlinear saturation (β). When comparing to the CGM, Eq. (1.25), the TSTM temperature equation Eq. (1.29) removes the logarithmic temperature gradient proportionality from the anomalous flux term, though still indirectly present through γ. C is a parameter absorbing the spatial scale of the turbulence, somewhat analogous to standard mixing length arguments, i.e. C is defined by [28]

$$\langle\tilde{T}^2\rangle = C^2\langle E\rangle\langle T\rangle^2, \tag{1.30}$$

where $\tilde{T}$ is the temperature fluctuations when expressing the turbulent heat flux

$$\Gamma_{\text{anomalous}} = \tanh(\gamma)\sqrt{\langle\tilde{T}^2\rangle\langle v_{\text{radial}}^2\rangle}, \tag{1.31}$$

v_{radial} being the fluctuating radial velocity component.

In the TSTM, turbulence itself becomes a transported quantity and spreads into the linearly stable regions of the plasma raising transport in these regions to high levels. Therefore the TSTM describes asymmetric radial spreading of turbulence. Also, the finite up-gradient anomalous transport in the subcritical region allows for modeling polarity reversal in the core. Nonlocality enters the system through the turbulence which introduces the possibility of nonlinear plasma responses caused by change in the turbulence level originating from spatially separated regions.

1.8 Strategy

An analytical proof of existence and uniqueness of solutions for the CGM and TSTM is beyond the scope of this thesis, if at all possible. In the following, the ‘engineering approach’ is employed: step by step constructing and verifying numerical implementations of the models, while carefully testing the code during development.

Chapter 2 presents the setup and verification of the numerical scheme used to model the CGM and TSTM. In chapter 3 modeling results are presented and interpreted, while comparing to experiment for validatation of the radial profile CGM and TSTM implementations. Then, the CGM is extended to include the poloidal dimension, allowing for investigations of the impact of 2D dynamics on heat modulation and cold pulse simulations. Finally, chapter 4 presents an alternative approach, allowing arbitrary 2D geometries, more closely resembling reactor conditions.

The tasks are summarised in the following points:

1. Identify requirements for numerical scheme.
2. Verify the implementation of the scheme chosen.
3. Construct 1D models, compare to article results and experiment.
4. Construct 2D model, while carefully testing against the 1D code.
5. Conduct numerical experiments with the 2D code.

Chapter 2

Numerical Implementation

Within this chapter, the framework for the numerical investigations conducted in chapter 3 is defined and verified. The choice of numerical scheme is argued for in detail in section 2.1, whereafter the fundamentals of the developed code is laid out in section 2.2. Finally, the chosen scheme is implemented for an analytically solvable problem in section 2.3, thus verifying the implementation of the numerical scheme chosen.

2.1 Approach

The choice of programming language should reflect the intended use of the code developed. If the objective is to write efficient code for computationally heavy tasks, one is probably best off coding in either FORTRAN or C, as these languages allow efficient parallelisation via application programming interfaces such as Message Passing Interface (MPI) for CPU clusters, Compute Unified Device Architecture (CUDA) for Nvidia GPU clusters or Open Computing Language (OpenCL) for both. If, on the other hand, the goal is to test and experiment with less computationally demanding numerical implementations, choosing a high-level language — with simpler syntax and easy visualisation of results — both reduces debugging time and makes the parameter fitting process less painful. MATLAB is one such language, and is therefore the language of choice in this work.

A numerical scheme suitable for the CGM and TSTM is now established. Before treating numerical time-integration, the spatial discretisation of the CGM and TSTM is discussed. The finite difference method (FDM) is chosen, as finite difference discretisation in space is fairly straight-forward, thus reducing the development time better spent on implementing the CGM and TSTM. The discrete FDM approximations to the spatial derivates appearing in the models are called *spatial stencils*.

2.1.1 Spatial Stencils

Several choices of spatial stencils are available, some more appropriate than others, depending on the problem at hand. Consider the 1-dimensional (1D) radial part of the CGM and TSTM. If information comes from both directions a central approximation, taking information from both sides of the radial grid point evaluated, is appropriate. This is the case during the heating process to steady-state and during heat modulation of the ICRH source in the plasma, as the origin of heat waves is located inside the domain, such that heat waves propagate towards the plasma centre for $\rho < \rho_{\mathrm{ICRH}}$ and towards the plasma edge for $\rho > \rho_{\mathrm{ICRH}}$. If, on the other hand, information comes from one direction an upwinding scheme might better resolve the dynamics, the Euler methods shown in table 2.1 being the simplest. This is the case during cold pulse experiments, as the plasma is brought to steady-state and then exposed to a sudden temperature drop at the edge.

The simplest way to approximate the derivative of a function $s(x)$ at the ith grid point, x_i, is via one of the Euler methods, shown in table 2.1. These are

Table 2.1: Euler discretisation methods

`Forward Euler method`	$\Delta^{\mathrm{fE}} s(x_i) = \frac{s(x_{i+1})-s(x_i)}{\delta x}$
`Backward Euler method`	$\Delta^{\mathrm{bE}} s(x_i) = \frac{s(x_i)-s(x_{i-1})}{\delta x}$

motivated by the standard definition of the derivative as the limiting value as $\delta x \to 0$. Taylor expanding $s(x)$ around the ith grid point, x_i, yields

$$s(x_i+\delta x) = s(x_i) + \delta x \left.\frac{\mathrm{d}s}{\mathrm{d}x}\right|_{x_i} + \frac{1}{2}\delta x^2 \left.\frac{\mathrm{d}^2 s}{\mathrm{d}x^2}\right|_{x_i} + \frac{1}{3!}\delta x^3 \left.\frac{\mathrm{d}^3 s}{\mathrm{d}x^3}\right|_{x_i} + \mathcal{O}\left(\delta x^4\right), \quad (2.1)$$

$$s(x_i-\delta x) = s(x_i) - \delta x \left.\frac{\mathrm{d}s}{\mathrm{d}x}\right|_{x_i} + \frac{1}{2}\delta x^2 \left.\frac{\mathrm{d}^2 s}{\mathrm{d}x^2}\right|_{x_i} - \frac{1}{3!}\delta x^3 \left.\frac{\mathrm{d}^3 s}{\mathrm{d}x^3}\right|_{x_i} + \mathcal{O}\left(\delta x^4\right), \quad (2.2)$$

where $\mathcal{O}\left(\delta x^4\right)$ satisfies

$$f(\delta x) = \mathcal{O}\left(\delta x^4\right) \qquad \text{as} \qquad \delta x \to 0,$$

for some function, $f(\delta x)$, if this function can be bounded [19],

$$|f(\delta x)| < K|\delta x^4| \text{ for all } \delta x \text{ sufficiently small,}$$

for some constant K. This implies, that $f(\delta x)$ should decay to 0 at least as fast as δx^4.

Note then, that by rearranging the terms in Eqs. (2.1-2.2), the error in the forward- and backward Euler schemes can be written,

$$\frac{s(x_i+\delta x)-s(x_i)}{\delta x} - \left.\frac{\mathrm{d}s}{\mathrm{d}x}\right|_{x_i} = \frac{1}{2}\delta x \left.\frac{\mathrm{d}^2 s}{\mathrm{d}x^2}\right|_{x_i} + \frac{1}{3!}\delta x^2 \left.\frac{\mathrm{d}^3 s}{\mathrm{d}x^3}\right|_{x_i} + \mathcal{O}\left(\delta x^3\right), \quad (2.3)$$

$$\frac{s(x_i)-s(x_i-\delta x)}{\delta x} - \left.\frac{\mathrm{d}s}{\mathrm{d}x}\right|_{x_i} = -\frac{1}{2}\delta x \left.\frac{\mathrm{d}^2 s}{\mathrm{d}x^2}\right|_{x_i} + \frac{1}{3!}\delta x^2 \left.\frac{\mathrm{d}^3 s}{\mathrm{d}x^3}\right|_{x_i} - \mathcal{O}\left(\delta x^3\right). \quad (2.4)$$

Eqs. (2.3-2.4) demonstrate, that the Euler methods are $1st$ order accurate approximations to $\frac{\mathrm{d}s}{\mathrm{d}x}$, meaning that the size of the error is roughly proportional to δx.

Subtracting Eq. 2.2) from Eq. (2.1) yields

$$s(x_i+\delta x)-s(x_i-\delta x)=2\delta x\left.\frac{\mathrm{d}s}{\mathrm{d}x}\right|_{x_i}+\frac{1}{3}\delta x^3\left.\frac{\mathrm{d}^3s}{\mathrm{d}x^3}\right|_{x_i}+\mathcal{O}\left(\delta x^5\right)$$

$$\Downarrow$$

$$\frac{s(x_i+\delta x)-s(x_i-\delta x)}{2\delta x}-\left.\frac{\mathrm{d}s}{\mathrm{d}x}\right|_{x_i}=\frac{1}{3}\delta x^2\left.\frac{\mathrm{d}^3s}{\mathrm{d}x^3}\right|_{x_i}+\mathcal{O}\left(\delta x^4\right). \tag{2.5}$$

The RHS in Eq. (2.5) is the error in the *central approximation* of $\frac{\mathrm{d}s}{\mathrm{d}x}$,

$$\Delta^{\mathrm{ca}}s(x_i)=\frac{1}{2}\left[\Delta^{\mathrm{fE}}s(x_i)+\Delta^{\mathrm{bE}}s(x_i)\right], \tag{2.6}$$

showing that this approximation is $2nd$ order accurate, i.e. the error is roughly proportional to δx^2. Due to its symmetry (preferable for steady-state and heat modulation numerical experiments) and $2nd$ order accuracy, the central approximation is chosen for approximating $1st$ order derivatives in this work.

Using Eqs. (2.1-2.2) to write

$$s(x_i+\delta x)+s(x_i-\delta x)-2s(x_i)=\delta x^2\left.\frac{\mathrm{d}^2s}{\mathrm{d}x^2}\right|_{x_i}+\frac{2}{4!}\delta x^4\left.\frac{\mathrm{d}^4s}{\mathrm{d}x^4}\right|_{x_i}+\mathcal{O}\left(\delta x^6\right)$$

$$\Downarrow$$

$$\Delta^{\mathrm{fE}}\Delta^{\mathrm{bE}}s(x_i)=\left.\frac{\mathrm{d}^2s}{\mathrm{d}x^2}\right|_{x_i}+\frac{2}{4!}\delta x^2\left.\frac{\mathrm{d}^4s}{\mathrm{d}x^4}\right|_{x_i}+\mathcal{O}\left(\delta x^4\right), \tag{2.7}$$

the $2nd$ order accurate $2nd$ order derivative stencil, Eq. (2.7), is derived. As for the $1st$ order derivative stencil Eq. (2.6), Eq. (2.7) is chosen for its symmetry and due its $2nd$ order accuracy, ensuring that $1st$ and $2nd$ order derivatives, $\frac{\mathrm{d}s}{\mathrm{d}x}$ and $\frac{\mathrm{d}^2s}{\mathrm{d}x^2}$, have the same order of accuracy.

Other stencils, briefly used in section 3.5, include the $3rd$ order accurate upwind stencil

$$\Delta^{\mathrm{uw}}s(x_i)=\frac{-s_{i+2}+6s_{i+1}-3s_i-2s_{i-1}}{6\delta x}, \tag{2.8}$$

for which Eq. (2.7) is used for the $2nd$ order derivative in simulations, and the $4th$ order accurate 5-point $1st$ order- and $2nd$ order derivative stencils,

$$\Delta^{\mathrm{5p}}s(x_i)=\frac{-s_{i+2}+8s_{i+1}-8s_{i-1}+s_{i-2}}{12\delta x}, \tag{2.9}$$

$$\Delta^{\mathrm{5p2}}s(x_i)=\frac{-s_{i+2}+16s_{i+1}-30s_i+16s_{i-1}-s_{i-2}}{12\delta x^2}. \tag{2.10}$$

Having established a numerical scheme for the spatial discretisation, the next step is to deal with numerical time-integration.

Many distinct time-dependent partial differential equation (PDE) numerical schemes are found in the literature. They can be divided in two subgroups; *explicit* and *implicit* schemes. Each equation of an *explicit* system (one for each spatial grid point) gives one of the unknown quantities for the next time step directly in terms of the known quantities at the present time step. However, if each equation involves more than one of the next time step unknowns, one must solve a set of simultaneous equations to obtain the quantities at the next time step. Such schemes are *implicit.* Explicit and implicit schemes exhibit different stability characteristics. In order to illustrate these differing characteristics, it is instructive to define the concept of *absolute stability.*

2.1.2 Regions of Absolute Stability

Consider the test equation,

$$\frac{\mathrm{d}s}{\mathrm{d}t} = \lambda s. \tag{2.11}$$

The notion of *absolute stability* is based on this linear equation, although a study of absolute stability for a method yields information that is typically directly useful in determining an appropriate time step for nonlinear problems as well [19].

With the *explicit* forward Euler scheme, Eq. (2.11) becomes

$$\begin{aligned} \frac{s_{n+1} - s_n}{\delta t} &= \lambda s_n, \\ &\Downarrow \\ s_{n+1} &= [1 + \delta t\lambda] s_n, \end{aligned} \tag{2.12}$$

with the time step size denoted by δt and s_n denoting the solution at time step number n. Defining the forward Euler scheme *stability function* for the test problem Eq. (2.11),

$$\mathfrak{S}(\delta t\lambda) \equiv 1 + \delta t\lambda,$$

Eq. (2.12) can be written

$$s_n = [\mathfrak{S}(\delta t\lambda)]^n s_0. \tag{2.13}$$

Note then, that the condition $s_n \to 0$ as $n \to \infty$ is fulfilled if $|\mathfrak{S}(z)| < 1$, with the, in general, complex number $z = \delta t\lambda$. For the test case, Eq. (2.11), the *region of absolute stability* is thus the region *inside* the unit circle with centre at $z = -1$.

For the *implicit* backward Euler method,

$$\frac{s_{n+1} - s_n}{\delta t} = \lambda s_{n+1},$$
$$\Downarrow$$
$$s_{n+1} = \frac{1}{1 - \delta t \lambda} s_n, \tag{2.14}$$

the stability function is

$$\mathfrak{S}(z) \equiv \frac{1}{1 - z},$$

i.e. the condition of absolute stability $|\mathfrak{S}(z)| < 1$ is fulfilled in the region *outside* the unit circle with centre at $z = 1$.

Thus, the explicit forward Euler method displays absolute stability *inside* a limited region, while the implicit backward Euler method display absolute stability *outside* a limited region.

2.1.3 Choosing a Time-integration Scheme

Often, all stability worries can be avoided by choosing an implicit scheme for the PDE one wishes to solve [31]. For ordinary differential equations (ODEs), methods with high order of accuracy, such as Runge-Kutta, good accuracy for small amounts of labour can be achieved. But this efficient method has no true counterpart for PDEs. For CGM and TSTM modeling, a stiffly stable time-solver is needed, due to the inherent stiffness originating from the discontinuous Heaviside (CGM) and γ (TSTM) functions. Physically, stiffness can be said to signify the fact that the solution has constituents which independently evolve on widely varying time-scales [6]. Many numerical methods, including all explicit methods, are unstable (in the sense of absolute stability) for such systems, unless the time step is small relative to the time-scale of the rapid transient [19]. For the stiff CGM and TSTM, the latter time-scale is much smaller (in fact, the Heaviside and γ functions contain a discontinuous transition, though smoothened somewhat by the $[\kappa_T - \kappa_c]$ factor in the CGM, and $\tanh(\cdots)$ function in the TSTM) than the time-scale of the smooth turbulent energy and plasma temperature solutions sought computed.

In this work, the choice falls upon a scheme using MATLAB's `ode23tb` solver designed for time-integration of stiff problems. MATLAB's other time-integration functions have been tested, though `ode23tb` is found to outperform these in terms of stability and computation time for CGM and TSTM implementations. The `ode23tb` solver is an implementation of the trapezoidal rule second order backward difference formula (TR-BDF2), which is an implicit Runge-Kutta formula [26]. The computational load of the temporal and spatial resolutions required to numerically model with MATLAB the phenomena observed in fusion experiments is assessed to be within the range of reasonable wall times.

For the remainder of section 2.1, further details are given of the numerical scheme constructed and used for CGM and TSTM simulations.

2.1.4 Stiffly Stable Time Integrator for a Nonlinear PDE

In [1] the Crank-Nicolson method is used to model fusion plasma turbulent transport described by a diffusion equation. This method is widely used for solving the heat equation and similar PDEs numerically. Therefore, it appears suitable for the problem at hand. The Crank-Nicolson method is

- $2nd$ order accurate in time,
- implicit in time,
- numerically stable.

For the diffusion equation, Eq. (1.24), with constant χ, the Crank-Nicolson method can be shown to be A-stable, meaning that the stability function is such that the stability region determined by $\mathfrak{S}(z) < 1$ contains the entire left half ($\mathfrak{Re}(z) < 0$) of the complex z-plane. Explicit methods have finite stability regions [6], so in order to guarantee A-stability an implicit method must be used. The tradeoff for having A-stability at each time step iteration is thus, as mentioned earlier, that a system of nonlinear equations need to be solved simultaneously at each time step. The Crank-Nicolson method is based on the *implicit* trapezoidal rule, obtained by letting the forward Euler LHS equal the RHS average of the two Euler methods, Eqs. (2.12) and (2.14),

$$\frac{s_{n+1} - s_n}{\delta t} = \frac{1}{2}[f_{\mathrm{RHS}}(s_n, t_n) + f_{\mathrm{RHS}}(s_{n+1}, t_{n+1})],$$

$$\Downarrow$$

$$s_{n+1} = s_n + \frac{1}{2}\delta t[f_{\mathrm{RHS}}(s_n, t_n) + f_{\mathrm{RHS}}(s_{n+1}, t_{n+1})], \tag{2.15}$$

t_n denoting the time step value at time step number n, s_n being the solution at that time step, δt the time step size, and f_{RHS} represents the PDE RHS.

Applying the scheme represented by Eq. (2.15) to the test case Eq. (2.11), the stability function is readily derived to be

$$\mathfrak{S}(z) = \frac{1 + \frac{1}{2}z}{1 - \frac{1}{2}z}, \tag{2.16}$$

which shows the region of absolute stability is exactly the $\mathfrak{Re}(z) < 1$ part of the complex z-plane. The trapezoidal method does damp all decaying components, though rapidly varying components are damped only very mildly since $\mathfrak{S}(z) \to 1$ as $z \to -\infty$. In fact, for stiff nonlinear time-dependent PDEs, numerical instabilities are known to arise due to insufficient numerical damping from the Crank-Nicolson method, as e.g. round-off errors can accumulate and cause the numerical solution to diverge. It has been shown [6] for a number of non-linear PDE test cases – including an advection-diffusion-reaction – that these instabilities can be remedied with the TR-BDF2 scheme.

The TR-BDF2 scheme consists of two steps:

1. Advance s_n to $s_{n+\iota}$ using the $2nd$ order accurate trapezoidal rule,
2. then advance $s_{n+\iota}$ to s_{n+1} via the $2nd$ order accurate backward difference formula,

or put mathematically for the test problem, Eq. (2.11), with $z = \delta t \lambda$ [6]

1. $s_{n+\iota} - \iota\frac{z}{2}s_{n+\iota} = s_n + \iota\frac{z}{2}s_n$,
2. $s_{n+1} - \frac{1-\iota}{2-\iota}zs_{n+1} = \frac{1}{\iota[2-\iota]}s_{n+\iota} - \frac{[1-\iota]^2}{\iota[2-\iota]}s_n$.

The n-step backward difference formula (BDF) methods, of which the 2-step BDF constitutes step 2. in the TR-BDF2, are a sub-class of the linear multistep methods, and are very effective for stiff problems [19].

The Jacobians for the two TR-BDF2 steps are readily extracted from the LHSs,

$$\begin{aligned} J_{1st \text{ step}} &= I - \frac{1}{2}\iota z \\ J_{2nd \text{ step}} &= I - \frac{1-\iota}{2-\iota}z. \end{aligned}$$

Requiring $J_{1st \text{ step}} = J_{2nd \text{ step}}$ and solving for ι yields

$$\iota = 2 - \sqrt{2}. \tag{2.17}$$

Choosing this value ensures that only one matrix will have to be inverted at each time step, even though two implicit schemes are used. It has also been proven [3] that this value minimises the local truncation error (LTE), defined by replacing the numerical approximation of the solution with the true solution [19]. Thus, the global truncation error — the accumulation over all iterations of LTEs made in each iteration — is also limited.

Substituting the trapezoidal rule TR-BDF2 1*st* step,

$$s_{n+\iota} = \frac{2+\iota z}{2-\iota z} y_n,$$

into the 2-step BDF 2*nd* TR-BDF2 step yields,

$$s_{n+1} - \frac{1-\iota}{2-\iota} z s_{n+1} = \frac{1}{\iota[2-\iota]} \frac{2+\iota z}{2-\iota z} s_n - \frac{[1-\iota]^2}{\iota[2-\iota]} s_n$$

$$\Downarrow$$

$$s_{n+1} = -\frac{2z+4-2\iota z-2\iota+\iota^2 z}{[-2+\iota z][-z+\iota z+2-\iota]}. \tag{2.18}$$

With the ι step choice, Eq. (2.17), Eq. (2.18) becomes

$$s_{n+1} = \frac{\left[2+\sqrt{8}\right]\left[z+\sqrt{2}+1\right]}{\left[2+\sqrt{2}-z\right]^2} s_n. \tag{2.19}$$

The stability function can be read directly from this,

$$\mathfrak{S}(z) = \frac{\left[2+\sqrt{8}\right]\left[z+\sqrt{2}+1\right]}{\left[2+\sqrt{2}-z\right]^2},$$

for which $\mathfrak{S}(z) \to 0$ as $|z| \to \infty$. By definition then, the TR-BDF2 scheme is L-stable [19], a stronger numerical scheme feature than A-stability.

The TR-BDF2 scheme, Eq. (2.19), is used in this work for the time-integration of the CGM and TSTM, due to the 2-step BDF method's effectiveness for stiff problems, and the excellent stability properties of the TR-BDF2 scheme as a whole.

2.2 Code Structure

Two distinct implementations are developed, the first for both the CGM and TSTM, the second for the CGM only. The two implementations do share some characteristics, which will first be outlined. After these common features are explained, an account of both implementations is given in separate sections.

2.2.1 Common Implementation Characteristics

As established in section 2.1, the discretisation tools selected for the numerical implementations of the CGM and TSTM are as follows:

- Stencils represented in Eqs. (2.6-2.10) are used for the discretisation of spatial derivatives in the CGM and TSTM,
- MATLAB's `ode23tb` implementation of the TR-BDF2 scheme, Eq. (2.19), is used for numerical time-integration.

Both implementations include a main script initialising the necessary parameters, that calls a RHS script for the specified model, via `ode23tb`.

The radial grid over ρ is constructed as an input number of equidistant grid points, $\delta\rho$ apart, ranging from

$$\rho_{\text{first}} = \rho_{\text{centre}} + \frac{\delta\rho}{2} \qquad \text{to} \qquad \rho_{\text{last}} = \rho_{\text{edge}} - \frac{\delta\rho}{2},$$

i.e. with the plasma centre-ρ as the first array element and the plasma edge-ρ as the last array element.

At the boundaries of the spatial region considered, the use of ghost points at $\rho_{\text{centre}} - \frac{\delta\rho}{2}$ and $\rho_{\text{edge}} + \frac{\delta\rho}{2}$ ensures that boundary conditions are satisfied at ρ_{centre} and ρ_{edge}, while at the same time evaluating the model in question at the physical grid points ρ_{first} and ρ_{last}. Furthermore, displacing $\rho_{\text{centre}} = 0$ by $\frac{\delta\rho}{2}$, the singularity issue at ρ_{first}, arising from taking the divergence in cylindrical coordinates, is avoided. Alternatively, one-sided stencils could have been applied at the physical boundaries, though this has been deselected in the present work such that the solution is treated the same way at all physical grid points.

The time resolution is controlled by the relative error tolerance, `RelTol`, and the absolute error tolerance, `AbsTol`, parameters. For each time step, `ode23tb` estimates the local error in the ith component of the solution. This error, ϵ_i, is a function of the specified `RelTol` and `AbsTol`, and is required to be less than or equal to the acceptable error [26],

$$|\epsilon_i| \leq \max\left(\mathsf{RelTol} \cdot |s_i|, \mathsf{AbsTol}_i\right), \tag{2.20}$$

s_i denoting the solution at the ith grid point, and AbsTol_i denoting the absolute tolerance at the ith grid point. Thus,

- RelTol roughly controls the number of correct digits in all solution components, except those smaller than AbsTol_i, i.e. it is a measure of the error relative to the size of each solution component, while

- AbsTol is a threshold below which the value of the ith solution component is unimportant, i.e. this vector determines the accuracy when the solution approaches zero.

Initial turbulent energy (TSTM) and plasma temperature (CGM & TSTM) profiles are provided for the `ode23tb` function, and all parameters are collected in a structure which is input as a function argument to the CGM and TSTM RHS MATLAB functions, called by `ode23tb` in the main script.

2.2.2 Explicit Derivatives Approach

In the first implementations developed, the spatial discretisation stencils for the derivatives are flexible. From the main script, the stencils Eqs. (2.6-2.7) or (2.9-2.10) can be chosen between, and for cold pulse simulations, Eq. (2.8) can be activated just before the pulse is initiated at the plasma edge. All derivatives in the CGM/TSTM RHS MATLAB functions are written out explicitly, so – using e.g. the same central approximation scheme at all grid points – boundary conditions are only needed for the turbulent energy (TSTM) and plasma temperature (CGM & TSTM) when applying derivative stencils.

Only the direct derivatives implementation is extended to 2D.

2.2.3 Direct Derivatives Approach

In the direct derivatives approach, there is no spatial derivative stencil flexibility. Only the $1st$ order derivative central approximation, Eq. (2.6), is employed and hardcoded throughout the CGM RHS MATLAB function. Consequently, the dynamics is modeled in a more ‘convective-like’ way, than in the explicit derivatives code employing the ‘diffusive-like’ $2nd$ order derivative stencil, Eq. (2.7). Contrary to the explicit derivatives implementation, all derivatives are here taken directly, such that e.g. the heat flux quantity is first evaluated and *thereafter* the divergence is taken of this quantity. Rather than employing one-sided $1st$ order derivative stencils at the plasma bounds, suitable ghost points are added for the additional quantities numerically differentiated.

2.3 Verification

During development of numerical models, it may be helpful to device a way to test the building blocks separately. In this way, the origin of issues can be identified and fixed, one at a time, rather than debugging the whole numerical model after completion. Identifying and testing these building blocks is the process of *verification.* In the present section a suitable development phase is identified and tested by recalling the origin of the dynamics modeled by the CGM and TSTM, in order to extract an equation that represents the underlying physics, and which can be solved analytically. This allows for error estimation of the numerical implementation

The diffusive nature of thermal dynamics in general supports the choice of the diffusion equation, Eq. (1.24), for this task.

2.3.1 Analytical Solution to the Diffusion Equation

For completeness, the diffusion equation, Eq. (1.24), is here restated,

$$\frac{\partial \mathcal{E}}{\partial t} = \nabla \cdot \chi \nabla \mathcal{E} + \mathcal{S}. \tag{2.21}$$

Assuming χ is constant in time and space, the radial part in cylindrical coordinates reads,

$$\mathcal{E}_t = \frac{1}{\rho} \chi \mathcal{E}_\rho + \chi \mathcal{E}_{\rho\rho} + \mathcal{S}. \tag{2.22}$$

Now, the solution to the homogeneous diffusion equation (i.e. $\mathcal{S} = 0$) is found.

Using separation of variables, i.e. setting $\mathcal{E}(t, \rho) = \mathcal{T}(t)\mathcal{R}(\rho)$, Eq. (2.22) is rewritten;

$$\frac{\mathcal{T}_t}{\mathcal{T}} = \frac{\mathcal{R}_\rho}{\rho \mathcal{R}} + \frac{\mathcal{R}_{\rho\rho}}{\mathcal{R}} + \mathcal{S}.$$

Thus, the LHS is now independent of ρ and the RHS is independent of t. Each side must then be equal to a constant, here denoted $-\lambda^2$. Thereby, Eq. (2.22) has been split in the two equations;

$$\mathcal{T}_t + \lambda^2 \mathcal{T} = 0 \tag{2.23}$$

$$\rho \mathcal{R}_{\rho\rho} + \mathcal{R}_\rho + \lambda^2 \rho \mathcal{R} = 0. \tag{2.24}$$

Eq. (2.23) has the solution,

$$\mathcal{T}(t) = c_1 \exp\left(-\lambda^2 t\right),$$

where c_1 is a constant that remains to be determined. In order to be interpreted as a diffusive physical quantity when multiplied with $\mathcal{R}(\rho)$, $\mathcal{T}(t)$ must be bounded and real for all times, $t > 0$. Requiring that $\lambda > 0$, the minus sign in the exponential will ensure that $\mathcal{T}(t)$ decays, rather than grows, as a function of t.

Eq. (2.24) is the parametric form of Bessel's equation of order 0, with λ as the parameter [2]. Eq. (2.24) is $2nd$ order and homogeneous, so two linearly independent solutions are needed in order to obtain the general solution. These are called Bessel functions of order 0 of the first and second kind, in the following denoted $J_0(\lambda\rho)$ and $Y_0(\lambda\rho)$, respectively. The general solution to Eq. (2.24) is then

$$\mathcal{R}(\rho) = c_2 J_0(\lambda\rho) + c_3 Y_0(\lambda\rho).$$

Let Eq. (2.22) be defined on the interval $\rho \in [0, R]$. As $\mathcal{T}(t)$ is required to be bounded for all $t > 0$, also $\mathcal{R}(\rho)$ must be bounded for $0 \leq \rho \leq R$. Since $Y_0(x)$ is not bounded for $x = 0$ [34], $c_3 = 0$ is required for $\mathcal{R}(\rho)$ to be bounded at $\rho = 0$. $J_0(x)$ satisfies the homogeneous Neumann condition at $x = 0$, so having $c_3 = 0$ corresponds to imposing the homogeneous Neumanns boundary condition on $\mathcal{R}(\rho)$ at $\rho = 0$, which coincides with the boundary condition imposed in the plasma centre in the 1D CGM and TSTM models. At the other bound, $\rho = R$, the homogeneous Dirichlet boundary condition is chosen, since $J_0(\lambda R) = 0$ leads to $\lambda_n = \mathfrak{r}_n/R$, $\mathfrak{r}_n$ being the nth root of $J_0(x)$. An inhomogeneous Dirichlet condition would results in a number of $\mathfrak{r}_n$ values that are not zeroes of $J_0(x)$, and therefore harder to find.

Setting $C = c_1 c_2$, the full solution can be written

$$\begin{aligned}\mathcal{E}(t,\rho) &= \mathcal{T}(t)\mathcal{R}(\rho)\\ &= \sum_{n=1}^{\infty} C_n \exp\left(-\lambda_n^2 t\right) J_0(\lambda_n \rho).\end{aligned}$$

Initially, at $t = 0$, this is

$$\mathcal{E}(0,\rho) = \sum_{n=1}^{\infty} C_n J_0(\lambda_n \rho). \tag{2.25}$$

In order obtain C_n, theorem 2, section 4.8 in [2], is restated below,

Theorem 1 *If $\mathcal{E}$ is piecewise smooth in $\rho \in [0, R]$, then $\mathcal{E}$ has a Bessel series expansion of order p on interval $\rho \in [0, R]$ given by*

$$\mathcal{E}(\rho) = \sum_{n=1}^{\infty} J_p(\lambda_{pn}\rho) \frac{2}{R^2 J_{p+1}(\mathfrak{r}_{pn})} \int_0^R \mathcal{E}(\rho) J_p(\lambda_{pn}\rho)\rho \, \mathrm{d}\rho,$$

where λ_{pn} are the scaled positive zeroes of the Bessel function J_p. In the interval $\rho \in [0, R]$, the series converges to $\mathcal{E}(\rho)$ where $\mathcal{E}$ is continuous and converges to the average $\frac{\mathcal{E}(\rho+)+\mathcal{E}(\rho-)}{2}$ at the points of discontinuity.

Using this theorem, Eq. (2.25) can be written

$$\mathcal{E}(0, \rho) = \sum_{n=1}^{\infty} J_0(\lambda_{pn}\rho) \frac{2}{R^2 J_{p+1}(\mathfrak{r}_{pn})} \int_0^R \mathcal{E}(0, \rho) J_p(\lambda_{pn}\rho)\rho \, \mathrm{d}\rho,$$

thus identifying C_n. Including the time-dependence, the full solution is

$$\mathcal{E}(t, \rho) = \sum_{n=1}^{\infty} \exp\left(-\lambda_n^2 t\right) J_0(\lambda_n \rho) \frac{2}{R^2 J_1^2(\mathfrak{r}_n)} \int_0^R \mathrm{d}\rho \;\; \mathcal{E}(0, \rho) J_0(\lambda_n \rho)\rho, \qquad (2.26)$$

with boundary conditions:

$$\left. \frac{\partial \mathcal{E}(t, \rho)}{\partial \rho} \right|_{\rho=0} = 0 \quad \& \quad \mathcal{E}(t, R) = 0 \qquad (2.27)$$

The initial condition chosen for simulations, is

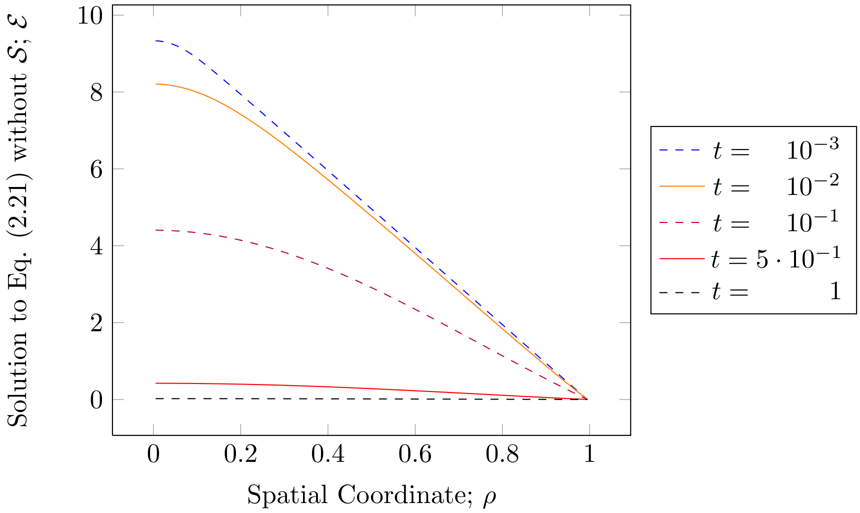
$$\mathcal{E}(0, \rho) = -10\rho + 10 \qquad \rho \in [0, R] \qquad (2.28)$$

The interval bound, R, is set to 1.

2.3.2 Comparison to Numerical Solution

For the comparison between the analytical and numerical solution to the diffusion equation, Eq. (2.21), the code presented in appendix A has been developed. The explicit derivatives approach, outlined in section 2.2.2, is used. Fig 2.1 (a) shows the analytical solution at various times, and Fig 2.1 (b) shows the deviation between analytical and numerical solutions. Fig. 2.1 shows, that the deviation between the analytical and numerical solution is initially greatest at the ends of the spatial interval, $\rho \in [0, 1]$. This spatial dependence is in line with the expectation:

(a) Time evolution of $\mathcal{E}(t,\rho)$, Eq. (2.21)



(b) Deviation between analytical and numerical solutions

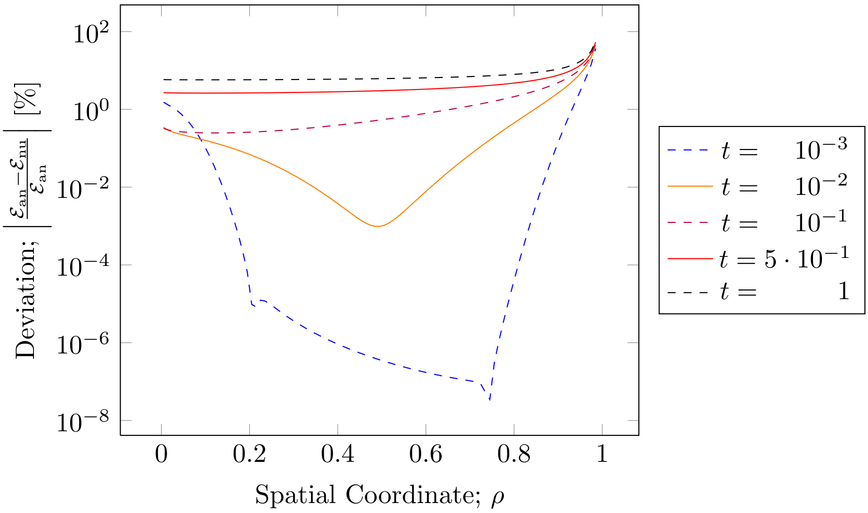
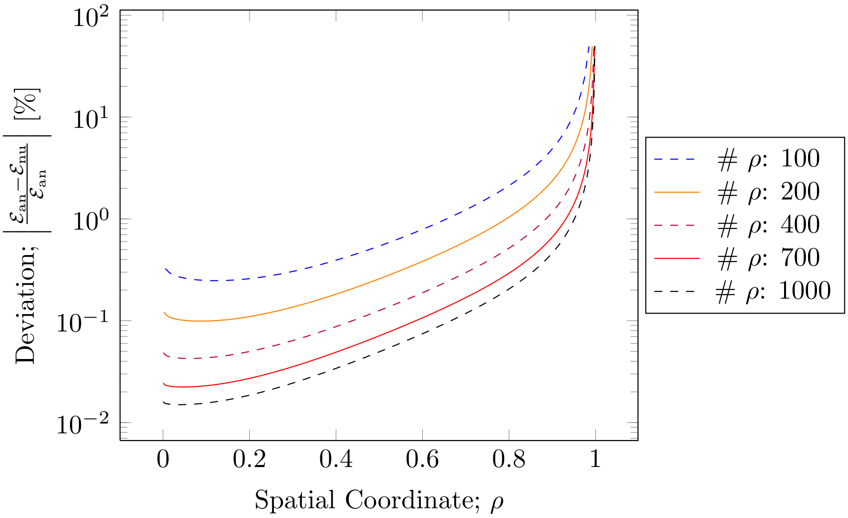


Figure 2.1: (a) shows the time evolution of $\mathcal{E}$, described by Eq. (2.21), with boundary conditions (2.27) and initial profile (2.28). In (b), the absolute value of the deviation between the analytical and numerical solutions is shown for the to (a) corresponding times.

(a) Deviation as function of spatial resolution; RelTol = AbsTol = 10^{-4}



(b) Deviation as function of temporal resolution; # ρ: 100

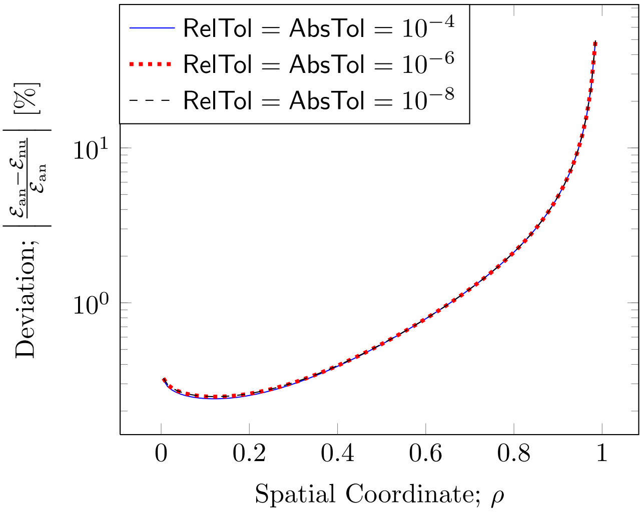


Figure 2.2: Deviation between analytical and numerical solutions to $\mathcal{E}$, described by Eq. (2.21), with boundary conditions (2.27) and initial profile (2.28), evaluated at time $t = 10^{-1}$. (a): Deviation as function of spatial resolution. (b): Deviation as function of temporal resolution.

- the magnitude of the temporal rate of change is greatest at $\rho = 0$ (greatest distance $\Delta\mathcal{E}$ to steady-state, i.e. the 0-profile),
- for $\rho \rightarrow 1$, $\mathcal{E}_{\text{an}} \rightarrow 0$, making the deviation at $\rho = 1$ undefined, which is why the deviation (very large value) at this spatial bound is left out of Fig. 2.1.

The rather large ($> 1\%$) deviations observed in Fig. 2.1 (b) for $t = 5 \cdot 10^{-1}$ and $t = 1$ may seem disturbing, however, the $\mathcal{E}$-profiles in Fig. 2.1 (a) corresponding to these times are observed to be almost 0 on the whole interval, which explains why the relative deviation, $\sim \frac{1}{\mathcal{E}_{\text{an}}}$, rises.

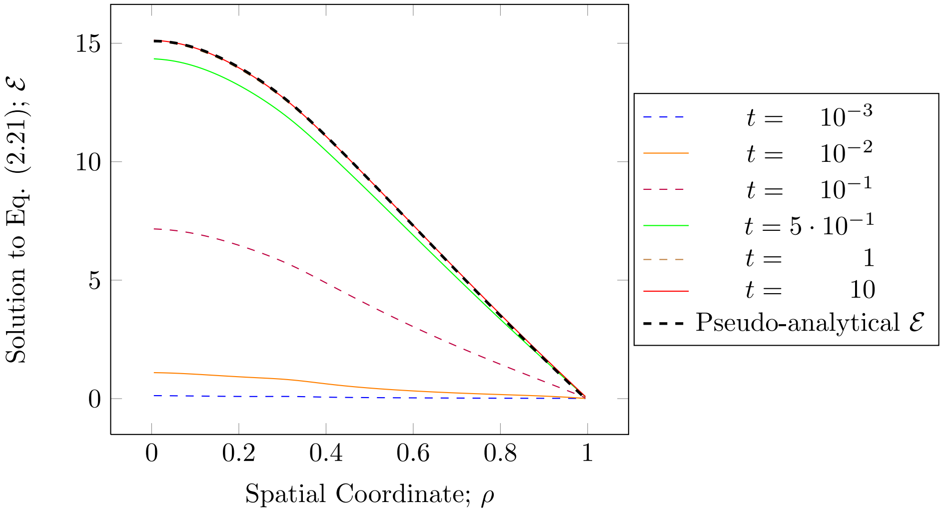
Fig. 2.2 shows the deviation between analytical and numerical solutions at $t = 10^{-1}$ for various spatial and temporal resolutions. Fig. 2.2 (a) shows that the deviation decreases as the number of ρ grid points, $\#\ \rho$, is increased. Fig. 2.2 (b) indicates that the ode23tb time-integrator used for the numerical solution of Eq. (2.21) performs well, as no significant $\mathcal{E}$ improvement is detected going from $\mathsf{RelTol} = \mathsf{AbsTol} = 10^{-4}$ to $\mathsf{RelTol} = \mathsf{AbsTol} = 10^{-8}$.

In a fusion experiment, the heating – represented by $\mathcal{S}$ for Eq. (2.21) assuming this equation describes the dynamics well – brings the plasma to a certain spatial profile, depending on the physics underlying the transport, where the magnitude of the heat flow into the plasma equals the heat flow lost to the surroundings, flowing across the plasma edge. Neglecting the fluctuations always present in real life experiments, this spatial profile is constant in time and is termed the *steady-state.*

To keep the verification process closely linked to the modeling of the CGM and TSTM, the source profile for these, Eqs. (3.1-3.3), is used for the solution to the inhomogeneous Eq. (2.21). However, the Gaussian dependence of the ICRH part complicates derivation of an analytical expression. Therefore, the numerical implementation is compared against a pseudo-analytical steady-state solution to Eq. (2.21), namely MATLAB's boundary value problem solver, bvp4c [26]. The results are presented in Fig. 2.3. (a) shows the convergence from a flat initial profile at 0 towards the pseudo-analytical steady-state. In (b), the deviation between pseudo-analytical- and numerical solutions are observed to decrease as a function of the spatial resolution. As is the case with the solution to the homogeneous Eq. (2.21) (i.e. $\mathcal{S} = 0$), deviations seem to depend mainly on spatial resolution, as no significant deviation change is oberved varying the temporal resolution.

It should be noted, that the deviations between analytical and numerical results are not only due to pure numerical LTE and rounding errors, occuring for e.g. the trapezoidal integration method used to perform the numerical integration in

(a) Steady-state convergence



(b) Spatial resolution scan

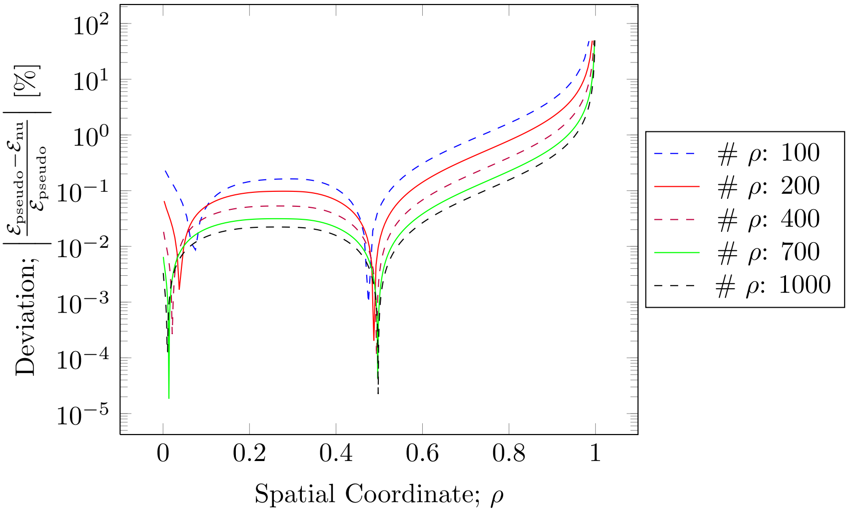


Figure 2.3: (a) shows the steady-state convergence of Eq. (2.21), with $\mathcal{S}$ given by Eqs. (3.1-3.3), starting from a flat initial profile at 0. (b) shows the dependence on spatial resolution of the deviation between the pseudo-analytical steady-state solution obtained with MATLAB's `bvp4c` and the numerical solution, with boundary conditions (2.27) and evaluated at time $t = 10$.

Eq. (2.26). Besides such errors of numerical nature, deviations are also subject to embedded errors originating from the fact, that, for practical purposes, only a finite number of terms can be included in the infinite sum appearing in the analytical solution, Eq. (2.26). However, the magnitude and decrease in deviation observed in Figs. 2.2 (a) and 2.3 (b) as a function of spatial resolution indicate that the embedded errors in the analytical solution are insignificant.

Based on the results presented in this section, the setup devised is assessed feasible for numerical implementation of the CGM and TSTM. Though the TR-BDF2 method's theoretical efficient handling of stiffness remains to be demonstrated in practice, the implementation of time-integrator and spatial stencils using ghost points works.

CHAPTER 3

Investigations & Interpretation of Results

For the study conducted, only the electron species is considered, as ion diagnostics are inferior in most fusion devices [29]. Furthermore, as stated in section 1.5, the electron density profile is assumed frozen and flat in simulations, since no density modulation due to the modulated electron cyclotron resonance heating (ECRH) could be detected within the precision [15] for at the Axially Symmetric Divertor Experiment (ASDEX) Uprade. In the present work, the data fitted for originates from an experiment conducted at the Joint European Torus (JET), though the flat electron density assumption justified by ASDEX Upgrade results is assumed to translate to JET.

In ASDEX Upgrade the propagation of ECRH induced heat wave pulses and cold pulses at the plasma edge are not identical, but exhibit similarities in the form of instability to anomalous transport [24]. In such electron heating experiments electron heat transport is dominantly driven by trapped electron mode (TEM) instability. However, experiments in which TEM, ion temperature gradient (ITG), electron temperature gradient (ETG) and other instabilities all come into play are clearly the conditions relevant for ITER operation [38]. Perturbation experiments used to be conducted by exploiting magneto-hydrodynamic (MHD) events, however this technique has its drawbacks [24]. Therefore this work focuses on perturbative experiments carried out at JET with coupled ion/-

electron heating.

All experimental data is from JET shot 55809 in which the issues of electron temperature profile stiffness and nonlocality was investigated via off-axis ion cyclotron resonance heating (ICRH) modulation and cold pulse experiments for a low confinement mode (L-mode) deuterium plasma with 15-20% helium 3 [21]. ITER will run with a mix of deuterium and tritium, due the high reaction rate for these hydrogen isotopes, as observed in Fig. 1.1. However, recall from section 1.1 that experiments involving tritium are complicated as this isotope is radioactive. Helium 3 is not radioactive and the weight is very close to that of tritium. Therefore, the fuels used in JET shot 55809 are probably the approximation yielding results closest to actual ITER conditions. The steady-state plasma temperatures are in the range 0.1 to 5`keV`, 1`keV` being $\approx 1.16 \cdot 10^7$`Kelvin`.

In order to validate the numerical implementations, model results found in the literature should be reproduced and comparison should be made to data from JET shot 55809. The validation basis includes:

1. Steady-state temperature radial profile.
2. Heat modulation amplitude & phase radial profiles.
3. Cold pulse temperature temporal profile.

The chapter is organised as follows:

- Section 3.1: Definition of source profile used for investigations.
- Section 3.2: Discussion of boundary- and initial conditions.
- Section 3.3: Modeling results for the TSTM as presented in [28].
- Section 3.4: Literature reference study.
- Section 3.5: Modeling based on the explicit derivatives approach.
- Section 3.6: Modeling based on the direct derivatives approach.
- Section 3.7: Poloidal cross-section modeling based on the direct derivatives approach.

3.1 Theoretical Heating Source Profile

In the ITER project, several different heating sources will be needed in order to reach sufficiently high temperatures for sustaining fusion processes. As mentioned in section 1.3, one such method is heating via neutral beam injection (NBI), which involves injecting a high-energy beam of fusion fuel atoms into the core of the plasma. The injected fast atoms become ionised and slowed down by the background plasma, thereby raising the overall plasma temperature. NBI heating is a very important option for steady-state operation of DEMO; the demonstration plant succeeding ITER [38].

In JET shot 55809, the basic plasma heating is done by NBI. However, from the fusion researcher's point of view, NBI has a major drawback: Localised heating is very hard, if not impossible to achieve. In order to conduct a heat modulation experiment suitable for comparison to radial profile numerical investigations in JET, NBI is supplemented by ICRH in the mode conversion scheme, chaneling energy to the electron species. A gyrotron launches microwaves with frequencies within the range of plasma species gyro frequencies – see e.g. Eq. (1.11) – into the plasma [22]. Due to the approximate frequency match there is a strong coupling between gyrotron microwave energy and plasma species, i.e. energy is transferred to the plasma with high efficiency. ICRH can be applied locally, such that temporal oscillation in temperature at some spatial point in the plasma becomes possible.

The radial source profile is approximated numerically by the mathematical relation,

$$\mathcal{S}(\rho) = \mathcal{NBI}(\rho) + \mathcal{A}(t) \cdot \mathcal{ICRH}(\rho). \tag{3.1}$$

In steady-state numerical experiments $\mathcal{A}(t)$ is constant in time, while for heat modulation numerical experiments it oscillates with frequency 15`Hz` (as in JET shot 55809 [21]) and an amplitude fitted to experimental data. $\mathcal{NBI}$ and $\mathcal{ICRH}$ are expressed mathematically as [29]

$$\mathcal{NBI}(\rho) = \frac{2}{3}\left[50 \cdot [1-\rho]^2 + 5\right], \tag{3.2}$$

$$\mathcal{ICRH}(\rho) = \frac{2}{3} \cdot 70 \cdot \exp\left(-[\rho - \rho_{\text{ICRH}}]^2 \cdot 17^2\right), \tag{3.3}$$

where the variable ρ is the normalised radial coordinate, defined in section 1.5. In JET shot 55809, the ICRH source is applied with peak intensity at $\rho_{\text{ICRH}} = 0.32$ in the radial profile, i.e. well off-axis (away from the plasma centre). This parameter, together with the time-dependent parameter $\mathcal{A}(t)$, has pronounced influence on the extend of the stiff middle region. Unless otherwise

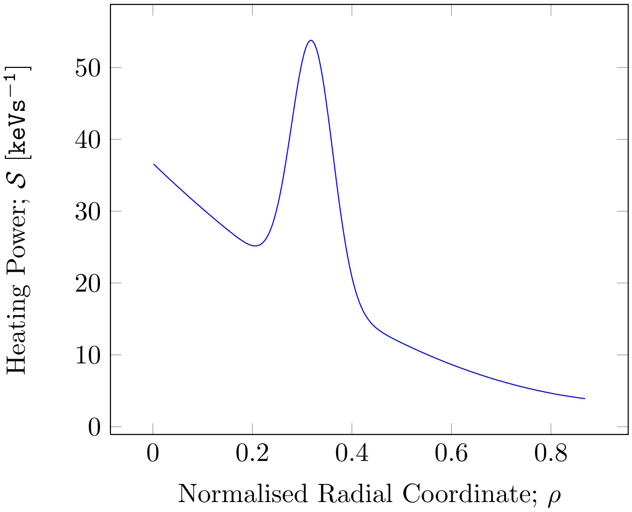


Figure 3.1: Source profile, Eqs. (3.1-3.3), used in CGM and TSTM numerical experiments.

stated, $\mathcal{A}(t) = 0.75$. The radial source profile, Eqs. (3.1-3.3), is plotted in Fig. 3.1. Increasing $\mathcal{A}(t)$ extends the stiff region, which, for a large enough value, could include the whole $\rho \in [\rho_{\mathrm{ICRH}}, \rho_{\mathrm{edge}}]$ region.

The radial source profile outlined in Eqs. (3.1-3.3) is used in all 1D investigations unless explicitly written otherwise.

3.2 Boundary- & Initial Conditions

Before proceeding, a discussion of the experimental conditions during JET shot 55809 is appropriate. In order to model the CGM and TSTM properly, one must first and foremost apply sufficient boundary conditions. From a mathematical point of view, the chosen boundary conditions must guarantee well-posedness, i.e. that:

1. A solution exists.
2. The solution is unique.
3. The solution depends continuously on the data.

Having e.g. homogeneous Neumann conditions at both plasma centre and edge of the radial temperature profile would result in an ill-posed problem; the source term forces the profile to grow without bound. This means, that in order to model the CGM and TSTM numerically, some boundary condition that bounds the profile must be introduced, even though the physics of the tokamak plasma do not dictate such constraints.

The boundary condition at the plasma centre is fixed by a mathematical/physical argument. Recall the fact that a 3D experiment is here modeled in 1 dimension; the measured JET shot 55809 radial temperature profile. Consequently, the PDE modeled must be assumed symmetric at the plasma centre. Mathematically, the profile of the physical quantity temperature should be continuous and smooth everywhere; having infinite temperature gradients imply instantaneous thermal transport which contradicts the theory of relativity. Therefore, a homogeneous Neumann condition is applied at the plasma centre, such that the temperature profile is smooth symmetrically kink-less across the plasma centre. However, no such symmetry argument can be used to fix the boundary condition at the plasma edge. In principle, any edge boundary condition satisfying the points stated below are appropriate. The chosen edge boundary condition should ensure that the numerically calculated profile

- is a fair match to the experimentally measured plasma temperature profile,
- displays finite temperature amplitude at the plasma edge as experimentally observed during heat modulation [21],
- as well as the edge boundary condition should depend explicitly on the edge temperature such that cold pulse data can be applied directly.

If modeled in a single run, the second and third point mutually excludes one another. In order to apply the cold pulse directly at the plasma edge, an inhomogeneous Dirichlet condition should be applied. This forces the temperature amplitude to `0keV` at the edge during heat modulation experiments. Finite amplitudes at the edge could be allowed if a Dirichlet condition is applied for a ρ point beyond the edge value, thus bounding the radial temperature profile. But this would be at the expense of performing cold pulse simulations.

An inhomogeneous Dirichlet condition based on the cold pulse data at $\rho = 0.87$ is chosen for the temperature at the plasma edge, i.e. vanishing edge amplitudes are accepted, even though it contradicts experiment.

Initial conditions are less important, since the temperature profile should converge towards the same steady-state, independent of the initial condition chosen. In JET shot 55809, the plasma is heated from low temperatures, so in principle any choice of initial profile below the steady-state should represent a fair match to the experimental conditions. A flat initial profile with the plasma edge Dirichlet temperature value is applied in all numerical experiments for which results are presented in section 3.5. This should approximate heating from noise and thus provide an estimate of the time it takes to reach steady-state after the NBI and ICRH sources have been switched on. In section 3.6 the initial spatial distribution of plasma temperature is chosen as the critical profile, thus reducing computation time.

3.3 Modeling the TSTM as Presented in [28]

Originally, the aim was to reproduce the results presented in [28]. The article presents the TSTM – details of which is given in section 1.7 – as a solution to the cold pulse problem stated in section 1.6, i.e. it explains with the same set of parameters the pulse dynamics originating from both ICRH modulation and laser ablation at the plasma edge.

In [28] the parameters listed in table 3.1 are used to obtain the numerical solution of Eqs. (1.26-1.29). The safety factor, q, used for simulations, is not specified in [28], so the spatially constant, $q = 2$, is here chosen [29]. In lack of better boundary conditions, homogeneous Neumann conditions are used at the plasma centre and edge for the turbulent energy [29].

The TSTM, Eqs. (1.28-1.29), is here implemented using the explicit derivatives approach, discussed in section 2.2.2. Derivatives,

Table 3.1: Parameters used in [28]

Parameter:	C	λ	χ_0	D_0	γ_0	β
Value:	0.5	1.7	0.6	35.0	0.5	1.0

```
% load derivatives
dE    = dEdr(E,1,1,data,ghostE);  d2E  = dEdr(E,1,2,data,ghostE);
dT    = dTdr(T,1,1,data,ghostT);  d2T  = dTdr(T,1,2,data,ghostT);
dTinv = dTdr(T,−1,1,data,ghostT); dT3h = dTdr(T,3/2,1,data,ghostT);
```

are called from within the RHS `ode23tb` argument function. The central approximation $1st$- and $2nd$ order derivative stencils, Eqs. (2.6-2.7), are implemented as,

```
% 1st & 2nd order derivatives of E
if order == 1
    out(2:end−1) = (in(3:end)−in(1:end−2))/(2*h);
    out(1)       = (in(2)−ghostE(1))/(2*h);
    out(end)     = (ghostE(2)−in(end−1))/(2*h);
elseif order == 2
    out(2:end−1) = (in(3:end)+in(1:end−2)−2*in(2:end−1))/hˆ2;
    out(1)       = (in(2)+ghostE(1)−2*in(1))/hˆ2;
    out(end)     = (ghostE(2)+in(end−1)−2*in(end))/hˆ2;
% 1st & 2nd order derivatives of T
Texp = in.ˆexpon;
if order == 1
    out(2:end−1) = (Texp(3:end)−Texp(1:end−2))/(2*h);
    out(1)       = (Texp(2)−ghostT(1)ˆexpon)/(2*h);
    out(end)     = (ghostT(2)ˆexpon−Texp(end−1))/(2*h);
elseif order == 2
    out(2:end−1) = (Texp(3:end)+Texp(1:end−2)−2*Texp(2:end−1))/hˆ2;
    out(1)       = (Texp(2)+ghostT(1)−2*Texp(1))/hˆ2;
    out(end)     = (ghostT(2)+Texp(end−1)−2*Texp(end))/hˆ2;
```

where `h` $= \delta\rho$, and `ghostE` and `ghostT` contain the plasma centre ghost points (homogeneous Neumann condition for both) as first element, and plasma edge ghost points (homogeneous Neumann for `E`, inhomogeneous Dirichlet for `T`) as second element, for turbulent energy and plasma temperature, respectively.

Next, the growth rate, γ, represented by Eqs. (1.26-1.27), is implemented as

```
% calculate gamma and its derivative
kappaT = −R*dT./T;
```

```
gam    = -0.1*lambda*sqrt(kappac-kappaT);
dgam   = -0.1*lambda*R*(d2T./T+dT.*dTinv)./sqrt(kappac-kappaT);
posgrowth = logical(imag(gam));
gam(posgrowth) = lambda*sqrt(kappaT(posgrowth)-kappac);
dgam(posgrowth) = -lambda*R*(d2T(posgrowth)./T(posgrowth)+dT(posgrowth) ...
                   .*dTinv(posgrowth))./sqrt(kappaT(posgrowth)-kappac);
```

such that the RHS can be evaluated,

```
% -------------------------------------------------------------------
spread = D0*(E.*dE./rho + dE.*dE + E.*d2E);
% -------------------------------------------------------------------
divtbs = -C*(E.*T.*tanh(gam)./rho + tanh(gam).*(dE.*T+E.*dT) ...
        + E.*T.*dgam.*(1-tanh(gam).^2));
% -------------------------------------------------------------------
divneo = 3/2*q*sqrt(q)*chi0*((dT./rho + d2T).*T.^(3/2) + dT3h.*dT);
% -------------------------------------------------------------------

span = 1:Nr; % span over physical grid points

dU_dt(span)    = spread+gam.*E-(gamma0+beta*E).*E;
dU_dt(Nr+span) = (divtbs + divneo);

S = NBI+amp*RF;

% add source
span = 1:Nr-1; % no source on edge (no material to heat)
               % ((and undermines T edge BC definition))

dU_dt(Nr+span) = dU_dt(Nr+span)+S(span);
```

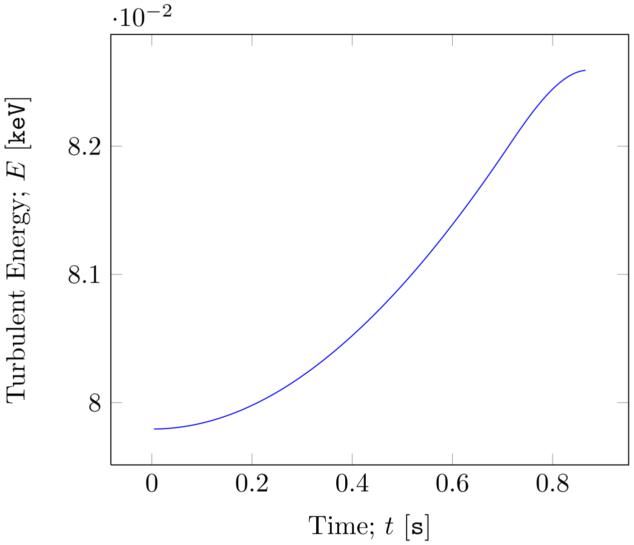
i.e. the `ode23tb` RHS function argument is one vector containing as the first $\# \rho$ elements the turbulent energy, E, and as the $\# \rho + 1$ to $2 \times \# \rho$ elements, the plasma temperature, T.

The full code can be viewed in appendix B.

The TSTM implementation is run from flat initial turbulent energy profile at 0.1`keV` and flat temperature profile at the edge inhomogeneous Dirichlet value. An inspection of the temperature time-trace for the ρ grid point at the plasma centre reveals that the temperature profile looks converged after approximately 100-200`ms`. The same analysis for the turbulent energy shows that it takes approximately 1 minute for the radial E profile to drop to its steady-state, an order of magnitude below the initial condition.

The turbulent energy profile and apparent steady-state plasma temperature profile are shown in Fig. 3.2, for the time 1`s`. It is clear, that the radial temperature profile in Fig. 3.2 (b) is a terrible fit to experiment. The profile can

(a) Radial turbulent energy profile



(b) Radial temperature profile

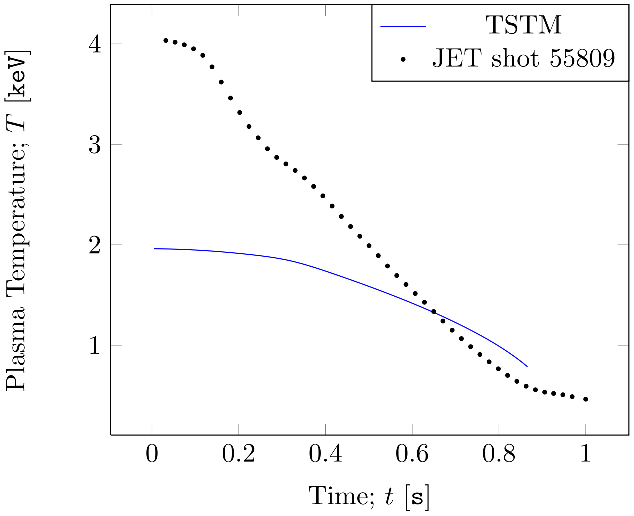


Figure 3.2: Turbulent energy profile (a) and steady-state temperature profile (b) predicted by the TSTM, Eqs. (1.26-1.29), after 1s. The TSTM parameters are listed in table 3.1. *Spatial resolution*: # ρ: 100. *Temporal resolution*: $\mathsf{RelTol} = \mathsf{AbsTol} = 10^{-4}$.

be adjusted upwards in magnitude, e.g. by decreasing q, but regardless of any fitting effort (parameters and source modification) it remains a ballooning-like profile, rather than displaying a core piedestal and close to straight line drop in the middle region towards the plasma edge, as in the experiment. This observed straight line region, the stiff region, is exactly what motivated the development of the CGM originally, so it should be present in the TSTM results as well. Across the stiff region, the logarithmic gradient is oscillating around the critical gradient. When greater, a large positive anomalous flux (physically turbulence developes) drains heat towards the edge, until the logarithmic gradient falls below the critical gradient again. The anomalous, neo-classical and total heat fluxes are shown in Fig. 3.3. The anomalous flux is vanishingly small and upgradient in the core and middle regions, only in edge region does it enter the super-critical regime, and that at much smaller magnitude than the neo-classical flux. The anomalous flux is proportional to γ, and a real time parameter analysis shows, that this quantity profile is indeed only positive in the edge region. Applied cold pulses at the plasma edge are observed to be strongly damped propagating inwards. Upadjusting the temperature profile does not change the essence of this; anomalous flux remains confined to the edge region and no set of parameters approximating the JET shot 55809 steady-state temperature profile has been found for which the TSTM, Eq. (1.28-1.29), produces anomalous flux – and thereby profile stiffness – in the middle region, $\approx \rho \in [0.3, 0.6]$. The vanishingly small anomalous flux explains why the temperature profile looks converged, though as long as the turbulent energy has not converged to its steady-state the temperature can not truly have reached steady-state, as the anomalous flux depends on the turbulent energy.

No amount of fitting and tweaking of parameters improves the situation. For the theoretical source profile, Eqs. (3.1-3.3), modification by

- boosting whole source profile,

- boosting source profile inversely proportional to ρ (keeping T_{edge} unchanged),

has been attempted unsuccessfully, as well as fitting the whole TSTM RHS `ode23tb` argument function by multiplying it with a factor.

Thus, the inferior TSTM results, visualised in Figs. 3.2-3.3, motivate the reference study, presented in section 3.4.

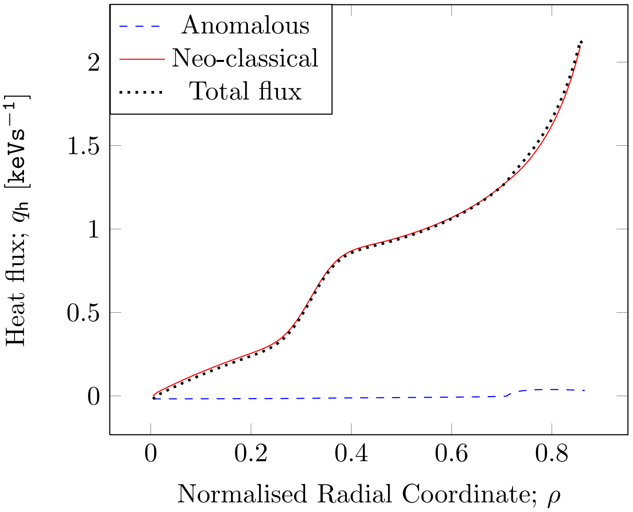


Figure 3.3: Anomalous, neo-classical and total heat fluxes obtained with implementation of the TSTM, Eqs. (1.26-1.29), after `1s`. The TSTM parameters are listed in table 3.1. *Spatial resolution*: # ρ: 100. *Temporal resolution*: RelTol = AbsTol = 10^{-4}.

3.4 Reference Study

The TSTM defined in [28], Eqs. (1.28-1.29), is an extension to the CGM, Eq. (1.25), a well-tested and succesful model as explained in section 1.5. Therefore, the results and references of [28] are here studied, in order to improve the understanding of the physics underlying the two models.

The CGM reference made in [28] is that of [9]. There, and in [24], the diffusion is governed by

$$\chi = \left[\chi_s \left[\frac{-R\nabla T}{T} - \kappa_c\right] \mathsf{H}\left(\frac{-R\nabla T}{T} - \kappa_c\right) + \chi_0\right] q^\nu \frac{T}{eB}\frac{\rho_s}{R}, \tag{3.4}$$

in which the dependence on additional parameters, e, B, and ρ_s, is present. The remaining parameters have the same meaning as in Eq. (1.25). The value $\nu = \frac{3}{2}$ is chosen, though ν was found to range between 1 and 2 [9], and this choice also applies in the CGM defined in [28] and section 1.5, Eq. (1.25). The magnitude of the electron charge, $e = 1.602 \cdot 10^{-19}\mathtt{C}$, B is the magnitude of the magnetic field, and $\rho_s = \frac{\sqrt{m_{\text{ion}} T}}{eB}$ is the Larmor radius.

The term,

$$\frac{T}{eB}\frac{\rho_s}{R}, \tag{3.5}$$

derives from the hypothesis of a turbulent transport characterised by an electrostatic gyroBohm scaling law, switched on above threshold, $\frac{-R\nabla T}{T} = \kappa_c$. The assumption of gyroBohm scaling relies on several recent turbulence simulations, however, substantial departure from gyroBohm scaling is found when the diamagnetic $\mathbf{E} \times \mathbf{B}$ (see sections 1.2-1.3) velocity shear rate is large or *when turbulence spreading takes place* [20]. This statement agrees with the observed removal of the $T^{\frac{3}{2}}$ factor from the anomalous flux term, when going from the CGM, Eq. (1.25), to the TSTM temperature equation, Eq. (1.29). However, as stated in [9] page 1353: "To simplify the calculation, the scaling law of this background diffusivity is supposed to be also gyroBohm (this means that the dependence on temperature is wrong when the background diffusivity is neoclassical)." In [9] this simplification allows for the construction of an analytical solution for the plasma temperature, followed by a detailed analysis. Nevertheless, having gyroBohm scaling, i.e. the factor Eq. (3.5), on the neo-classical term is physically unjustified.

Further reference studies reveals, that the CGM cold pulse data in [28] from 2008 – to which TSTM results are compared – originates from the 2002 article, [21]. The latter article defines the CGM diffusion coefficient as

$$\chi = \chi_0 + \lambda T^{\alpha}\left[-\frac{\nabla T}{T} - \kappa_c\right]^{\beta} \mathsf{H}\left(-\frac{\nabla T}{T} - \kappa_c\right), \tag{3.6}$$

with the choices $\alpha = \frac{3}{2}$ and $\beta = \frac{4}{5}$, and the parameter λ plays the role of χ_s for the anomalous diffusion. With this definition, the CGM reads

$$\frac{\partial T}{\partial t} = \nabla \cdot \left[\chi_0 + \lambda T^{\frac{3}{2}}\left[-\frac{\nabla T}{T} - \kappa_c\right]^{\frac{4}{5}} \mathsf{H}\left(-\frac{\nabla T}{T} - \kappa_c\right)\right]\nabla T + \mathcal{S}. \tag{3.7}$$

Notice, that for this definition, the $T^{\frac{3}{2}}$ factor, coming from the gyroBohm scaling, is only present in the anomalous flux term, as it should, according to [9]. Note also the missing dependence on the safety factor, q, and tokamak major radius, R. The value used for the critical gradient in [21], $\kappa_c = 1.8\mathtt{m}^{-1}$, is numerically the same as the corresponding dimensionless quantity used in [28], even though R, which do not appear in Eq. (3.7), is $\approx 3\mathtt{m}$ in JET, from which results are treated in the R-dependent CGM and TSTM from [28]. Indeed, as shown in Fig. 3.4, κ_c values approximately 3 times greater than that used in [28] are found in JET experiments. However, R is set to 1 for the results presented in section 3.3, so the κ_c value of 1.8 should be in the right range. The critical gradient is known to vary radially in Tore Supra [13], a fusion device situated

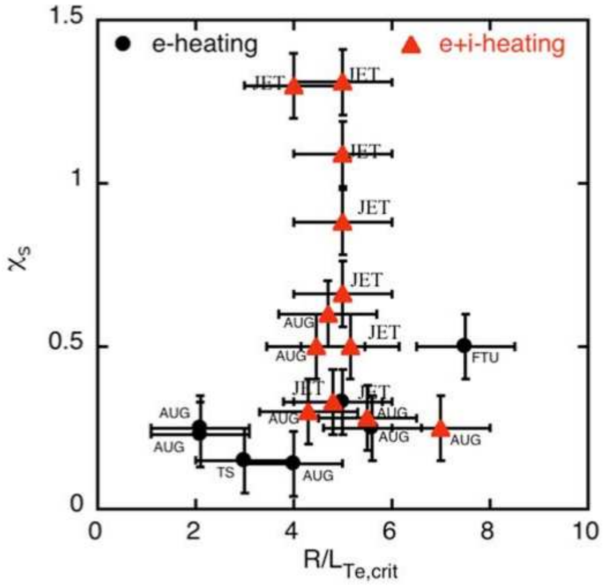


Fig. 12. Values of χ_s versus $R/L_{T_{e,\mathrm{crit}}}$ for pure electron heating and with NBI (e + i heating) in various machines.

Figure 3.4: The figure reproduced here is borrowed from [24], page 644. The anomalous diffusion parameter χ_s, is plotted against $\kappa_c = \frac{R}{L_{\mathrm{crit}}}$.

in Cadarache, France, though, as is the case in this study, it is usually assumed spatially constant in order to reduce the number of free parameters.

As already mentioned, the CGM definition, Eq. (3.7), is independent of the safety factor, q. This is a feature included later for both anomalous- and neo-classical flux terms, however, in [24] and [22] q is dropped from the neo-classical flux term (i.e. the χ_0 term, see [24] page 638 or [22], section II.C; page 1157). In fact, the whole idea of a safety factor dependence in the CGM is by no means final [25].

Including data for the electron density radial profile, as done in [24, 22], as well as spatially varying neo-classical diffusion, χ_0, and safety factor, q, would constitute a more quantitative study. Also, the source term in the diffusion equation, Eq. (1.24), should be reconstructed in a more direct manner, i.e. reconstruction based on time-traces of the various power sources used in the experiments modeled, rather than having the theoretical form, Eqs. (3.1-3.3). Such a theoretician-like approach is bound to be of limited use in quantitative studies [25]. In [24], the source profile is inferred from

$$\frac{\partial n_e}{\partial t} + \nabla \cdot \Gamma_e = S_e \tag{3.8}$$

$$\frac{\partial}{\partial t}\frac{3}{2}n_{e,i}T_{e,i} + \nabla \cdot \left(\mathbf{q}_{e,i} + \frac{5}{2}T_{e,i}\Gamma_{e,i}\right) = Q_{e,i} \tag{3.9}$$

where the density of electrons or ions is denoted $n_{e,i}$, the electron or ion temperature, $T_{e,i}$, $\Gamma_{e,i}$ & $\mathbf{q}_{e,i}$ the particle & heat fluxes, respectively, and $S_{e,i}$ & $Q_{e,i}$ are the particle and heat sources/sinks.

Several accounts of numerical results implementing such improvements exist. Rather than adding one such account of radial profile simulations, this work aims at investigating the impact of including the poloidal dimension in the simulation of tokamak thermal dynamics.

As the present reference study indicates, several empirical models based on a critical temperature gradient exist in the literature [23, 9, 22, 28, 21, 15]. To sum up the yield of the study, the distinct CGM definitions encountered are listed below.

$$[28]\ 2008: \frac{\partial T}{\partial t} = \frac{3}{2}\nabla \cdot \left[\chi_s\left[\frac{-R\nabla T}{T} - \kappa_c\right]\mathsf{H}\left(\frac{-R\nabla T}{T} - \kappa_c\right) + \chi_0\right] q^{\frac{3}{2}}T^{\frac{5}{2}}\frac{\nabla T}{T}$$

$$[22]\ 2006: \frac{\partial T}{\partial t} = \nabla \cdot \left[\chi_s\left[\frac{-2R\nabla T}{T} - \kappa_c\right]\mathsf{H}\left(\frac{-R\nabla T}{T} - \kappa_c\right) + q^{\frac{3}{2}}\chi_0\right]\frac{T}{eB}\frac{\rho_s}{R}\nabla T$$

$$[9]\ 2004: \frac{\partial T}{\partial t} = \nabla \cdot \left[\chi_s\left[\frac{-R\nabla T}{T} - \kappa_c\right]\mathsf{H}\left(\frac{-R\nabla T}{T} - \kappa_c\right) + \chi_0\right] q^{\nu}\frac{T}{eB}\frac{\rho_s}{R}\nabla T$$

$$[21]\ 2002: \frac{\partial T}{\partial t} = \nabla \cdot \left[\lambda T^{\frac{3}{2}}\left[\frac{-\nabla T}{T} - \kappa_c\right]\mathsf{H}\left(\frac{-\nabla T}{T} - \kappa_c\right) + \chi_0\right]\nabla T$$

Due to the physically unjustified gyroBohm scaling on the neo-classical flux term in Eqs. (1.25) and (1.29), and the questionable safety factor dependence for both anomalous- and neo-classical flux terms, the numerical investigations conducted in section 3.5 are based on the CGM diffusion coefficient definition used in e.g. [15, 23, 21], Eq. (3.7), and the gyroBohm and q dependencies of the neo-classical term in the TSTM temperature equation, Eq. (1.29), are removed.

Thus, the TSTM is recast as

$$\frac{\partial E}{\partial t} = \nabla \cdot D_0 E \nabla E + \gamma E - [\gamma_0 + \beta E]E, \tag{3.10}$$

$$\frac{\partial T}{\partial t} = \nabla \cdot [-CET\tanh(\gamma) + \chi_0 \nabla T]. \tag{3.11}$$

in which also the factor of $\frac{3}{2}$ has been dismissed, as new C and χ_0 values are needed anyway.

3.5 Radial Profile Investigations

For the numerical investigation results presented in this section, the CGM, Eq. (3.7), and TSTM, Eqs. (3.10-3.11), are implemented using the explicit derivatives approach outlined in section 2.2.2.

CGM results are presented in sections 3.5.1-3.5.4, whereafter TSTM results are discussed in section 3.5.5. The code used to generate results presented in this section can be viewed in appendix B.

3.5.1 Steady-State

In [21] the CGM, Eq. (3.7), is modeled with parameters $\kappa_c = 1.8\mathtt{m}^{-1}$, $\lambda = 2\mathtt{m}^3\mathtt{s}^{-1}\mathtt{keV}^{-\frac{3}{2}}$ (unit shown corresponds to $\beta = 1$) and $\beta = 0.8$. The missing information includes

- the neo-classical diffusion profile, χ_0,
- the source profile, $\mathcal{S}$,
- boundary- & initial conditions.

The source profile used in [21] is assumed to be close to the one represented by Eqs. (3.1-3.3). With this assumption – and no further knowledge of the modeling parameters used – a natural next step is to scan over the background diffusion constant χ_0 (assumed radially constant) and ICRH fitting parameter $\mathcal{A}$, while experimenting with the edge boundary condition guided by the framework laid out in section 3.2. To simplify this task further, the χ_0 value used in [28] is adopted as a first fitting estimate and as Fig. 3.5 indicates, this choice reproduces experimental data fairly well with $\mathcal{A}(t) = 1$. The CGM parameters used are listed in table 3.2.

Fig. 3.5 suggests that χ_0, the source profile and the boundary conditions used could be a good match to the actual experiment conditions, likely also with regard to ICRH heat modulation and cold pulse experiments. To assess the

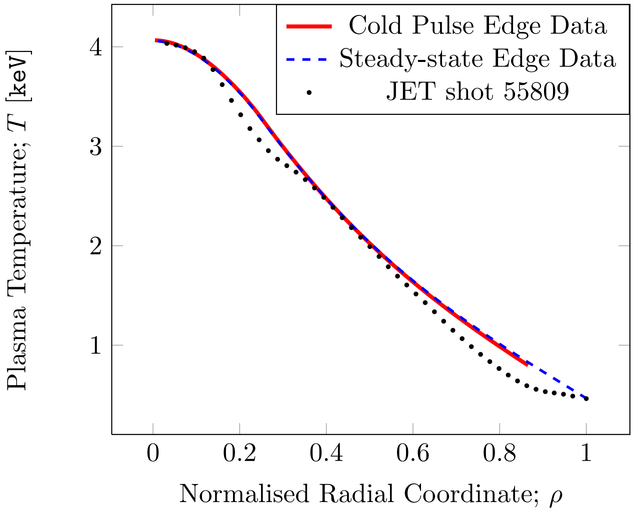


Figure 3.5: Steady-state (run time 2s) temperature profile predicted by the CGM, Eq. (3.7), compared to steady-state experimental data (JET shot 55809). The CGM parameters used are shown in table 3.2. Two distinct simulation results are shown: Temperature profile with inhomogeneous Dirichlet condition at the plasma edge from the JET shot 55809 cold pulse data (solid red), and temperature profile with JET shot 55809 steady-state data (dotted black) edge value as the inhomogeneous edge Dirichlet condition (dashed blue). *Spatial resolution*: # ρ: 100. *Temporal resolution*: RelTol = AbsTol = 10^{-4}.

quality of the implementation more carefully, the radial dependence of heat fluxes is analysed. In steady-state, the LHS of Eq. (3.7) is 0, meaning that

$$\mathcal{S} = -\nabla \cdot \left[\chi_0 + \lambda T^{\frac{3}{2}} \left[-\frac{\nabla T}{T} - \kappa_c\right]^{\frac{4}{5}} \mathsf{H}\left(-\frac{\nabla T}{T} - \kappa_c\right)\right] \nabla T \tag{3.12}$$

Cylindrical coordinate radial integration yields,

$$\begin{aligned}\int \mathcal{S}(\rho)\rho\partial\rho &= -\int \frac{1}{\rho}\left[\frac{\partial}{\partial\rho}\rho\left[\chi_0 + \lambda T^{\frac{3}{2}} \left[-\frac{\nabla T}{T} - \kappa_c\right]^{\frac{4}{5}} \mathsf{H}\left(-\frac{\nabla T}{T} - \kappa_c\right)\right] \nabla T\right]\rho\partial\rho \\ &= -\rho\left[\chi_0 + \lambda T^{\frac{3}{2}} \left[-\frac{\nabla T}{T} - \kappa_c\right]^{\frac{4}{5}} \mathsf{H}\left(-\frac{\nabla T}{T} - \kappa_c\right)\right] \nabla T \end{aligned} \tag{3.13}$$

If Eq. (3.13) is met, the system has reached steady-state. The Neumann condition at the plasma centre, $\rho = 0$, ensures the flux is always 0 there, as heat fluxes are proportional to the temperature gradient. Thus, all the heating generated

Table 3.2: The CGM parameters used in Eq. (3.7)

χ_0 $[\mathtt{s}^{-1}]$	λ $\left[\mathtt{s}^{-1}\mathtt{keV}^{-\frac{3}{2}}\right]$	κ_c	β
0.6	2.0	1.8	0.8

by the source profile, $\mathcal{S}(\rho)$, will have to leave the plasma across the edge. In Fig. 3.6 (b), the accumulated radial integral over the source (LHS of Eq. (3.13)) is plotted, along with the RHS of Eq. (3.13). Notice the departure from Eq. (3.13) in the middle- and edge regions. Fig. 3.6 (a) shows there is highly increased heat transport around $\rho = 0.25$. This is where the logarithmic gradient rises above the κ_c threshold allowing thermal turbulence to develop, in turn resulting in stiffness in that region. A strange jump occurs in the region where anomalous flux is triggered, and Fig. 3.6 (b) shows a significant deviation between the LHS and RHS of Eq. (3.13). Well above $\rho = 0.25$ — around $\rho = 0.4$ — the overall heat flux is observed to steadily drop.

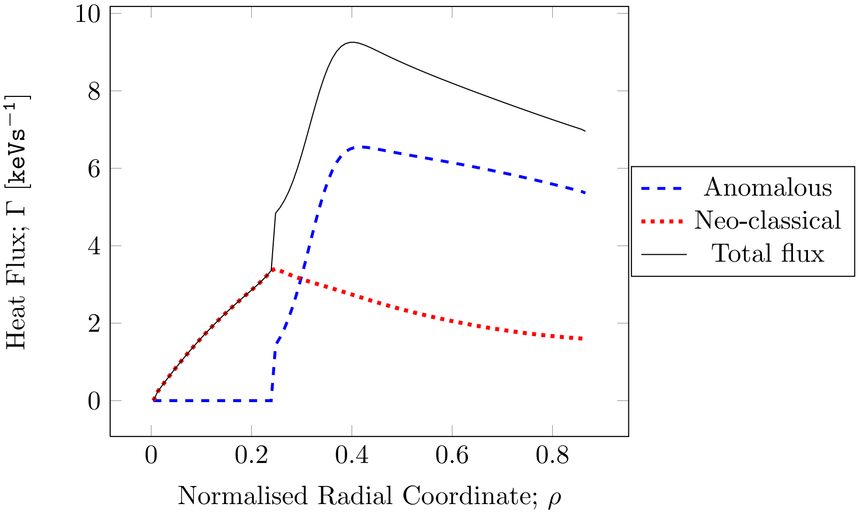
3.5.2 Dependence on Resolution

Even though the anomalous heat flux jump around $\rho = 0.25$ and the deviation between LHS and RHS of Eq. (3.13), shown in Fig. 3.6, are concerning, no error is identified in the CGM implementation. The fact, that the Eq. (3.13) LHS (the source heating) is below the RHS in Fig. 3.6 (b) indicates that more heat leaves the $\rho \in [0, 0.87]$ domain than enters via $\mathcal{S}$ at that particular time. This suggests, that $\rho\Gamma$ is oscillating around $\int \mathcal{S}(\rho)\rho\partial\rho$ in the steady-state. In order to test the CGM implementation further, the dependence on spatial and temporal resolutions is investigated.

Fig. 3.7 (a) shows the temperature time trace at the plasma centre, i.e. for the $\rho = \frac{\delta\rho}{2}$ grid point. There seems to be an instability around # ρ: 150 grid points, as the plasma centre time trace grows linearly with time. There is no reason to believe this growth only occurs at a spatial resolution of 150 grid points. Indeed, a plasma centre time trace growth is observed for simulations with a resolution of 117 radial grid points.

To see if the linear growth originates from the central approximation spatial stencils, Eqs. (2.6-2.7), used in the implementation, these are replaced by the 5-point stencils, Eqs. (2.9-2.10). In Fig. 3.8, CGM results for the central approximation and 5-point stencils are compared. Notice in (a), that the insta-

(a) Anomalous, neo-classical and total fluxes for CGM steady-state



(b) Accumulated integral over source profile compared the total flux

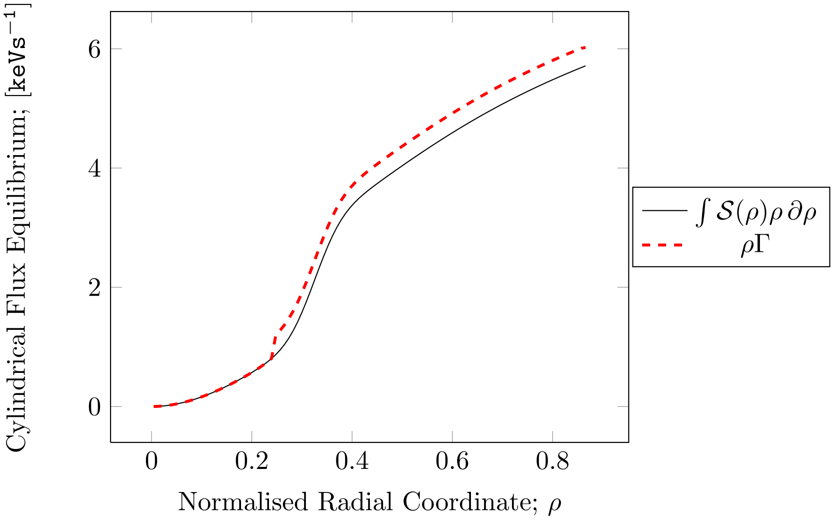
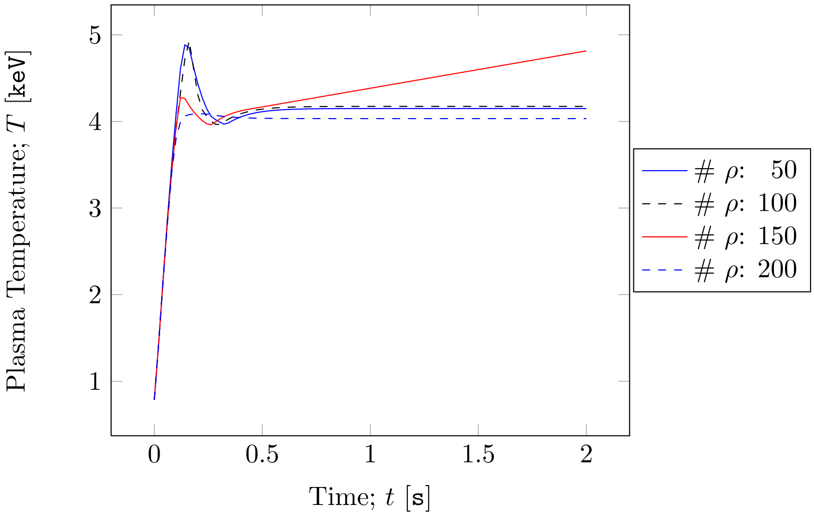


Figure 3.6: (a): The steady-state fluxes corresponding to the temperature profile shown in Fig. 3.5, with inhomogeneous Dirichlet value based on the JET shot 55809 cold pulse data. (b) Deviation between LHS and RHS in Eq. (3.13).

(a) Plasma centre time trace for various spatial resolutions



(b) Plasma centre time trace zoom around spatial resolution # ρ: 150

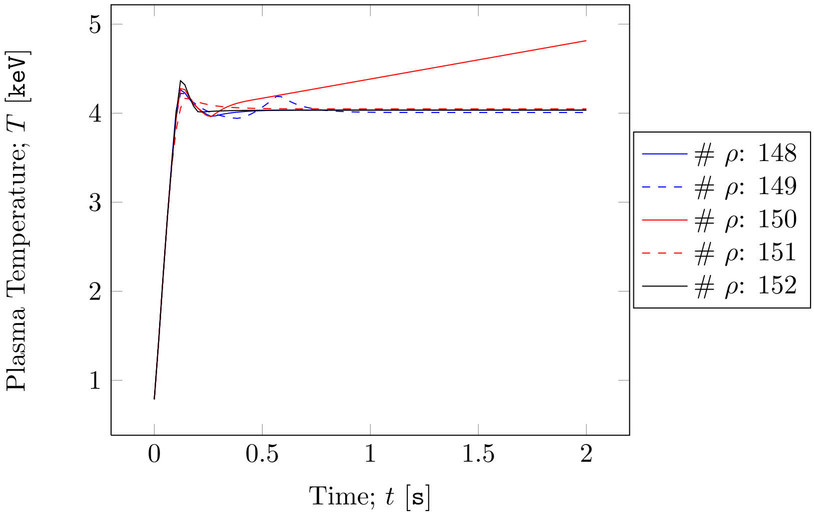
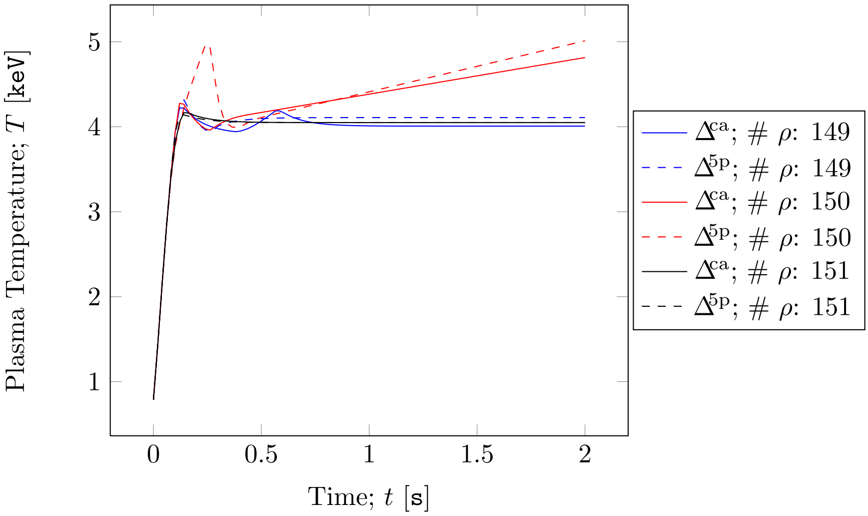


Figure 3.7: Plasma centre, i.e. $\rho = \frac{\delta\rho}{2}$, time traces for the CGM setup treated in section 3.5.1 for various spatial resolutions. *Temporal resolution*: $\mathsf{RelTol} = \mathsf{AbsTol} = 10^{-4}$.

(a) Plasma centre time trace for Δ^{ca} and Δ^{5p} for spatial resolutions in the vicinity of # ρ: 150



(b) Steady-state temperature profile for stencils Δ^{ca} and Δ^{5p}

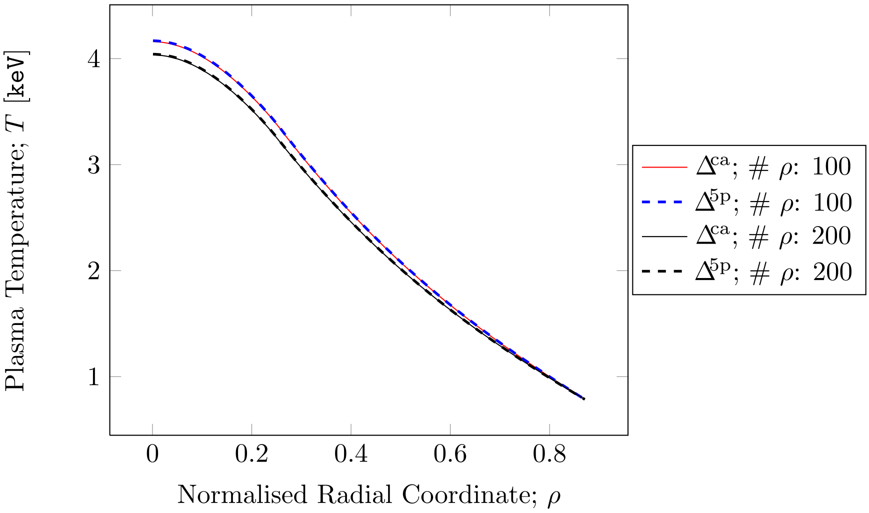


Figure 3.8: (a): Plasma centre, i.e. $\rho = \frac{\delta\rho}{2}$, time traces for the CGM setup treated in section 3.5.1 for spatial resolutions in the vicinity of the instability at # ρ: 150, for the central approximation spatial stencils Δ^{ca}, Eqs. (2.6-2.7), and the 5-point stencils Δ^{5p}, Eqs. (2.9-2.10). (b) Steady-state temperature profiles for Δ^{ca} and Δ^{5p} at # ρ: 100 and # ρ: 200, $\mathsf{RelTol} = \mathsf{AbsTol} = 10^{-4}$.

bility seems to grow more rapid for the $4th$ order accurate 5-point stencil CGM time trace, than for the $2nd$ order accurate central approximation CGM time trace, and also, the plasma centre temperature obtained with the 5-point stencil seems to 'overshoot' the steady-state severely around $t \sim 100 - 400\texttt{ms}$. Though, the slight instability at # ρ: 149 appears to be damped for the 5-point stencil CGM time trace results. Fig. 3.8 (b) shows good agreement between radial temperature profiles for the central approximation and 5-point stencils.

Since the linear growth seems to steepen with the order of accuracy of the FDM used to discretise derivatives in space, simulations with lower order methods might stabilise the implementation at these resolutions. The linear growths observed in the plasma centre time traces for $\mathsf{RelTol} = \mathsf{AbsTol} = 10^{-4}$ disappear from spatial resolutions # ρ: 117 and 150 when increasing the temperoral resolution to $\mathsf{RelTol} = \mathsf{AbsTol} = 8 \cdot 10^{-5}$.

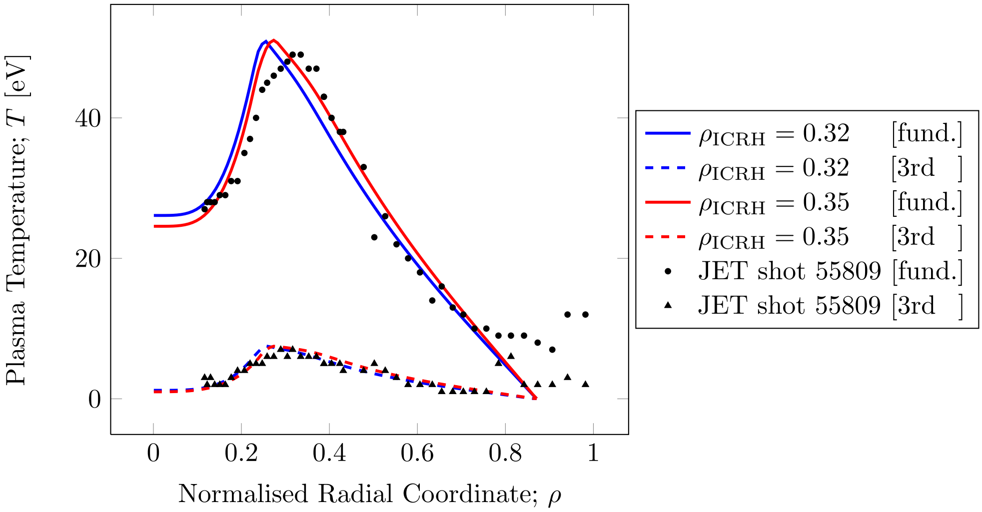
The exact origin of the linear growth observed in the plasma centre at certain resolutions remains unidentified, however, away from the aforementioned resolutions, simulations compare well to experimental data, as argued for in the following sections.

3.5.3 Heat Modulation

In JET shot 55809, the heat modulation experiment is conducted by periodically switching off and on the ICRH asymmetrical heating at $15\texttt{Hz}$, at a 50% duty cycle [21, 25]. This is here modeled using the CGM, Eq. (3.7), with the parameters listed in table 3.2. Recall from section 3.4, that spatially constant χ_0, λ and κ_c is at odds with observations [25], though chosen in order to reduce the number of free fitting parameters. Furthermore, the statement in section 3.4, that using a theoretical source profile, Eqs. (3.1-3.3), implies that agreement with experiment may be hard to find.

Yet, as the CGM heat modulation results in Fig. 3.9 indicate, these assumptions do not appear to be a grave over-simplification, and the theoretical source approximation, Eqs. (3.1-3.3) do allow for investigation of the impact of magnitude and spatial location of power deposition in the tokamak plasma. The $\rho_{\mathrm{ICRH}} = 0.32$ CGM heat modulation results displayed in Fig. 3.9 are a fair match to the experimental data. Shifting the ICRH peak intensity slightly, to $\rho_{\mathrm{ICRH}} = 0.35$, improves the CGM amplitude data match to the experimental JET data, at the expense of the CGM phase data. However, regardless of the ρ_{ICRH} modeled it is clear that the Dirichlet boundary condition at the plasma edge fails in reproducing the physics accurately. No physics-based argument suggests that such a forcing limitation at the plasma edge should have been present

(a) Amplitude profile: JET shot 55809 data compared to CGM as function of ρ_{ICRH}



(b) Phase profile: JET shot 55809 data compared to CGM as function of ρ_{ICRH}

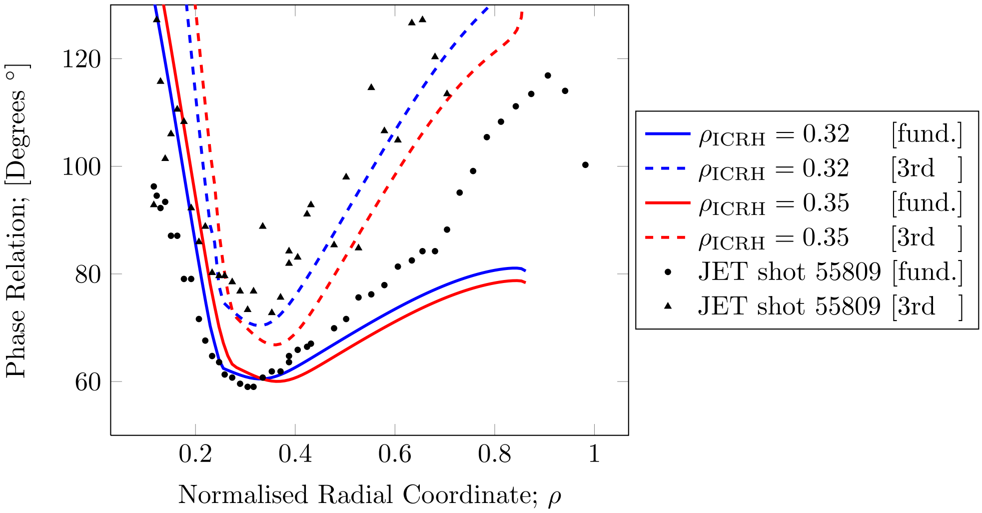
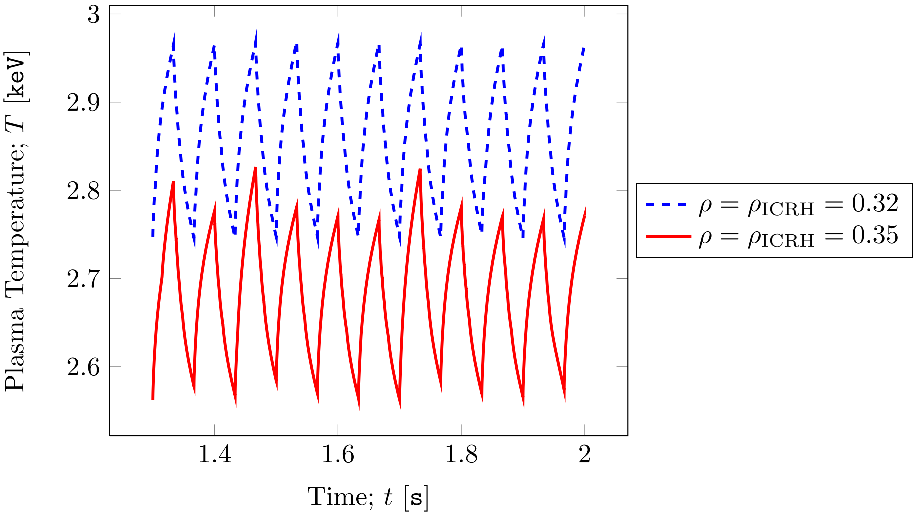


Figure 3.9: Heat modulation fundamental mode and 3rd harmonic amplitude and phase profiles predicted by the CGM, Eq. (3.7) with parameters listed in table 3.2, compared to JET shot 55809 data. *Source parameters*: $\mathcal{A}(t) = 1.1 \pm 1$. *Spatial resolution*: # ρ: 100. *Temporal resolution*: $\mathsf{RelTol} = \mathsf{AbsTol} = 8 \cdot 10^{-5}$.

(a) CGM time traces: $\rho = \rho_{\mathrm{ICRH}}$



(b) CGM DFT spectrum: $\rho = \rho_{\mathrm{ICRH}}$

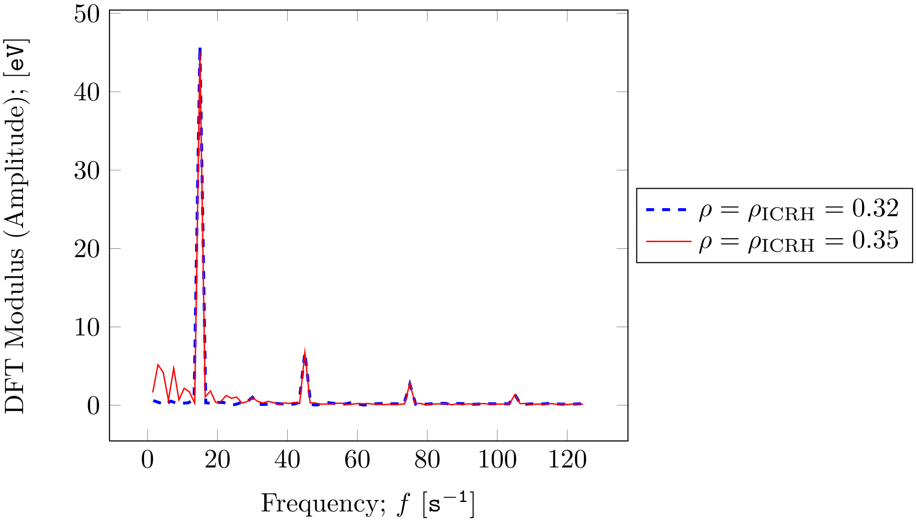


Figure 3.10: Time trace (a) and discrete fourier transform (DFT) modulus (b), for the CGM data shown in Fig. 3.9.

during JET shot 55809. Numerically however, the temperature profile must be bounded somehow as discussed in section 3.2. Since a homogeneous Neumann condition is required at the plasma centre, having a Dirichlet condition at the plasma edge bounds the temperature profile. As mentioned in section 3.2, one way to model finite amplitudes at the plasma edge is to extend the modeled radial interval using a Dirichlet condition at the edge of this extended interval, allowing finite amplitudes at the actual plasma edge located within the range of the 'physical' interval. This is not attempted here: In this work, the aim is to model both heat modulation and cold pulse dynamics, under the same physical conditions. In order to maintain a direct correspondence between these distinct numerical experiments, the boundary conditions employed should be as closely related as possible. Since a Dirichlet condition at the plasma edge is well-suited for numerical cold pulse experiments, plasma edge Dirichlet condition CGM results are presented in Fig. 3.9, without using the aforementioned method to obtain finite plasma edge temperature amplitudes. Likewise, the available JET cold pulse time trace plasma edge data is for $\rho = 0.87$, which is why the ρ-interval for the CGM heat modulation is $\rho \in [0, 0.87]$, rather than $\rho \in [0, 1]$ for which experimental heat modulation data is available.

The relationship between phase fundamental mode and third harmonic, in particular, is found to be very sensitive to variations in ρ_{ICRH}. For values $\rho_{\mathrm{ICRH}} = 0.30$ and $\rho_{\mathrm{ICRH}} = 0.34$ the third harmonic has lower phase than the fundamental mode, while Fig. 3.9 with $\rho_{\mathrm{ICRH}} = 0.32$ and $\rho_{\mathrm{ICRH}} = 0.35$ display a relationship much closer to experiment. However, from Fig. 3.9 (b) it is clear, that heat wave propagation is too slow toward the plasma centre, and too fast towards the plasma edge, when comparing to the JET shot 55809 phase data.

Due to the ICRH source temporal oscillations, the simulations providing the CGM heat modulation results presented in Fig. 3.9 exhibit a strong dependence on time resolution. In Fig. 3.10 temperature time traces at $\rho = \rho_{\mathrm{ICRH}}$ are plotted. These time traces seem to be near jibberish for $\mathsf{RelTol} = \mathsf{AbsTol} = 10^{-3}$, however, they look perfectly smooth for $\mathsf{RelTol} = \mathsf{AbsTol} = 10^{-4}$. Switching the spatial stencil from central approximation to 5-point stencil does not improve results significantly: Temporal resolution determines temperature profile time trace quality, and thereby the quality of the amplitude and phase profiles which are derived directly from the time trace data. See appendix B.5 for the code used to produce heat modulation results in Figs. 3.9 and 3.10.

Thus, the temperature time trace at ρ_{ICRH} must be inspected for every heat modulation run and found smoothly periodic in time, i.e. with a fairly stable amplitude, before the temporal resolution can be said to be sufficient and results accepted.

For the implementation used here, this requirement is estimated to be achieved at $\mathsf{RelTol} = \mathsf{AbsTol} = 8 \cdot 10^{-5}$, the lowest temporal resolution possible in this implementation for scanning over the ρ_{ICRH} displayed in Fig. 3.9. For increased temporal resolutions, the `ode23tb` time-solver fails to meet integration tolerances without reducing the step size below the smallest value allowed ($\sim 5 \cdot 10^{-17}\mathtt{s}$).

3.5.4 Cold Pulse

In Fig. 3.11 the time traces for 4 distinct plasma locations are shown. The JET shot 55809 cold pulse data measured at $\rho = 0.87$ is applied at this spatial location in the CGM simulation. Note the much faster cold pulse propagation obtained with the $3rd$ order accurate upwind spatial stencil, Eq. (2.8), as compared to the two symmetric stencils. The more diffusive $2nd$ order accurate upwind scheme is found to be unstable.

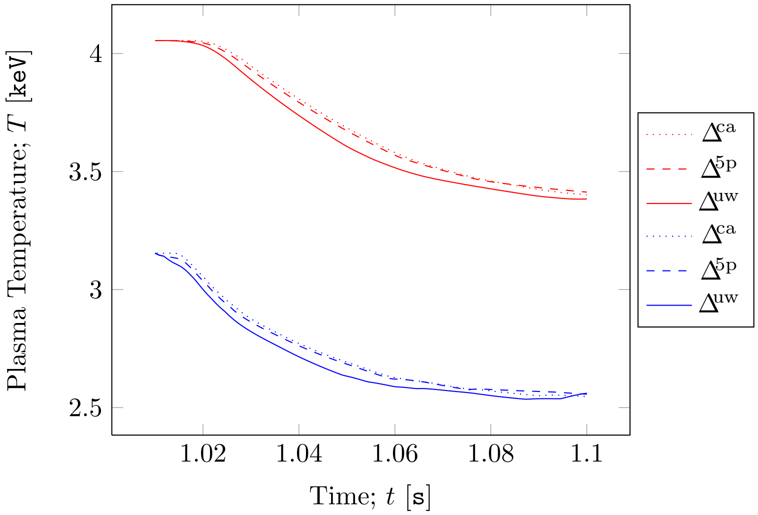
As Fig. 3.11 shows, the delay time for the cold pulse to propagate to the core depends strongly on the spatial stencil used. The delay times for a 30`eV` drop reaching $\rho = 0.11$, is in the range 14-27`ms`, depending on the spatial stencil used.

3.5.5 Discussion of TSTM Results

As for the plasma temperature in sections 3.5.1-3.5.4, the turbulent energy and temperature initial conditions are assumed spatially constant, keeping the initial turbulent energy profile level as an input parameter, whereas the initial temperature profile has the edge Dirichlet value for all simulations. Consequently, spatial gradients are initially 0 everywhere. This is in contrast to the case of some initial temperature scenario where large gradients must be computationally smeared out, while simultaneously adjusting to the applied source profile. Obviously, from a pure physics point of view, heating the plasma from much lower temperatures is what actually occurs in the JET experiments. However, assuming a flat initial turbulent energy profile and taking the steady-state edge Dirichlet data as a flat initial temperature profile should not make any difference, since the heat modulation and cold pulse JET experiments are conducted well after steady-state is reached.

In the explicit derivatives implementation (see section 2.2.2), the TSTM, Eqs.

(a) Cold pulse time traces for $\rho = 0.11$ (red) and $\rho = 0.30$ (blue)



(b) Cold pulse time traces for $\rho = 0.60$ (red) and $\rho = 0.87$ (blue)

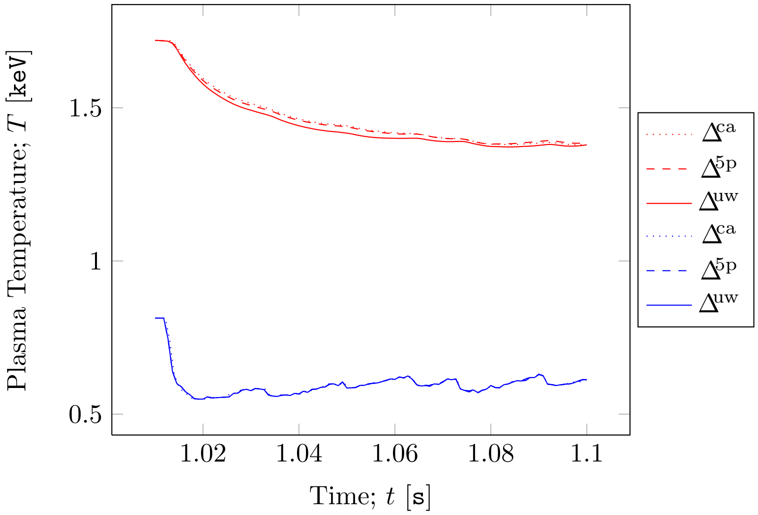


Figure 3.11: Cold pulse time trace for the CGM, Eq. (3.7), with parameters listed in table 3.2, for the central approximation, Eqs. (2.6-2.7), 5-point, Eqs. (2.9-2.10), and upwind, Eq. (2.8), spatial stencils. *Source parameters*: $\mathcal{A}(t) = 1.1$. *Spatial resolution*: $\#\ \rho$: 100. *Temporal resolution*: $\mathsf{RelTol} = \mathsf{AbsTol} = 10^{-5}$.

(3.10-3.11), are written,

$$\begin{aligned}
\frac{\partial E}{\partial t} &= \nabla \cdot D_0 E \nabla E + \gamma E - [\gamma_0 + \beta E]E \\
&= \frac{1}{\rho}\frac{\partial}{\partial \rho}\ \rho D_0 E \frac{\partial E}{\partial \rho} + \gamma E - [\gamma_0 + \beta E]E \\
&= D_0 \left[\frac{1}{\rho} E \frac{\partial E}{\partial \rho} + \left[\frac{\partial E}{\partial \rho}\right]^2 + E \frac{\partial^2 E}{\partial \rho^2} \right] + \gamma E - [\gamma_0 + \beta E]E \\
\frac{\partial T}{\partial t} &= \nabla \cdot [-CET \tanh(\gamma) + \chi_0 \nabla T] + \mathcal{S} \\
&= \frac{1}{\rho}\frac{\partial}{\partial \rho}\ \rho \left[-CET \tanh(\gamma) + \chi_0 \frac{\partial T}{\partial \rho} \right] + \mathcal{S} \\
&= -C \tanh(\gamma) \left[\frac{1}{\rho} ET + \frac{\partial E}{\partial \rho} T + E \frac{\partial T}{\partial \rho} \right] \\
&\quad - CET \frac{\partial \gamma}{\partial \rho} \left[1 - \tanh^2(\gamma)\right] + \frac{1}{\rho} \chi_0 \frac{\partial T}{\partial \rho} + \chi_0 \frac{\partial^2 T}{\partial \rho^2} + \mathcal{S},
\end{aligned}$$

for the radial dimension.

The numerical implementation of turbulent energy and temperature derivatives is as follows:

```
% load derivatives
dE    = dEdr(E, 1,1,data,ghostE); d2E  = dEdr(E, 1 ,2,data,ghostE);
dT    = dTdr(T, 1,1,data,ghostT); d2T  = dTdr(T, 1 ,2,data,ghostT);
dTinv = dTdr(T,−1,1,data,ghostT); dT3h = dTdr(T,3/2,1,data,ghostT);

% switch stencil
if     t>1.01 && fdm ==2 && sim_mode == 3 % cold pulse

    % upwind stencil
    dE    = dEdrUpwind(E,    1,data,ghostE);
    d2E   = dEdrUpwind(E,    2,data,ghostE);
    dT    = dTdrUpwind(T, 1 ,1,data,ghostT);
    d2T   = dTdrUpwind(T, 1 ,2,data,ghostT);
    dTinv = dTdrUpwind(T,−1 ,1,data,ghostT);
    dT3h  = dTdrUpwind(T,3/2,1,data,ghostT);

elseif t>1.01 && fdm ==3 && sim_mode ~= 1 % heat modulation & cold pulse

    % 5−point stencil
    dE    = dEdr5point(E,    1,data,ghostE);
    d2E   = dEdr5point(E,    2,data,ghostE);
    dT    = dTdr5point(T, 1 ,1,data,ghostT);
    d2T   = dTdr5point(T, 1 ,2,data,ghostT);
    dTinv = dTdr5point(T,−1 ,1,data,ghostT);
    dT3h  = dTdr5point(T,3/2,1,data,ghostT);
end
```

This allows for easy stencil flexibility, controlled by the parameter fdm, set in the main script. The optional $3rd$ order accurate upwind (fdm=2) and $4th$ order accurate 5-point (fdm=3) stencils are loaded in the if construct. The former is only used for cold pulse simulations, the latter for both heat modulation and cold pulse simulations. As Fig. 3.11 indicates, the upwind method yields slightly faster cold pulse propagation speeds – as expected from theory

Coding derivatives of quantities like γ and the anomalous flux directly would be straigthforward, were it not for their dependence on derivatives of turbulent energy and temperature. For the γ case, its derivatives then involves ghost points adjacent to the γ ghost points themselves *if the same spatial stencil should be used at all physical points*, so γ and the anomalous flux are here implemented with explicit derivatives of E and T. In this way only ghost points for turbulent energy and temperature are needed for the evaluation of these quantities, and the same stencils can be used at all grid points. γ and its derivative are coded as

```
kappaT = −dT./T;
gam    = −0.1*lambda*sqrt(kappac−kappaT);
dgam   = −0.1*lambda*(d2T./T+dT.*dTinv)./sqrt(kappac−kappaT);
posgrowth = logical(imag(gam));
gam(posgrowth)  = lambda*sqrt(kappaT(posgrowth)−kappac);
dgam(posgrowth) = −lambda*(d2T(posgrowth)./T(posgrowth)   ...
                  +dT(posgrowth).*dTinv(posgrowth))       ...
                  ./sqrt(kappaT(posgrowth)−kappac);
```

and thus $\frac{\partial \gamma}{\partial \rho}$ depends explicitly on the $1st$- and $2nd$ order derivative of temperature. The term describing turbulence spreading is written

```
spread = D0*(E.*dE./rho + dE.*dE + E.*d2E);
```

For the temperature equation, the divergence of the anomalous flux is

```
divtbs = −C*(E.*T.*tanh(gam)./rho + tanh(gam).*(dE.*T+E.*dT) ...
            + E.*T.*dgam.*(1−tanh(gam).^2)                  );
```

i.e. it depends on temperature derivatives through γ and $\frac{\partial \gamma}{\partial \rho}$. The background diffusion term is implemented as

```
divneo = chi0*((dT./rho + d2T);
```

Combining these parts constituting the TSTM system of coupled equations, Eqs. (3.10-3.11), yields

```
dU_dt(span)    = spread + gam.*E – (gamma0 + beta*E).*E;
dU_dt(Nr+span) = divtbs + divneo + NBI + amp*RF;
```

As mentioned in section 3.4, the CGM cold pulse data appearing in Fig. 3 [28] is originally modeled in [21]. In [21] there is no gyroBohm dependence on the χ_0 term, as is the case in [28]. This supports the gyroBohm-free TSTM definition, Eqs. (3.10-3.11), implemented here.

Initial TSTM simulations for the parameters listed in table 3.1 have been flawed by numerical instability at certain parameters. Real time analysis of the radial γ profile indicates that the discontinous transition from subcritical to turbulent transport pose a numerical challenge for the `ode23tb` solver. At some temporal resolutions, instabilities would build in the sub- and supercritical transition ρ-region and then die out, while for higher temporal resolutions the γ profile would diverge. The RHS of the TSTM temperature equation, Eq. 3.11, depends on the gradient of γ, which is undefined at the discontinuity, and indeed commenting out the term involving this gradient results in fast simulations. The issue of undefined gradients arises in the CGM definition as well. However, contrary to the CGM case which involves the gradient of the Heaviside function, the gradient of the γ function is not 0 below the threshold in the TSTM case.

Straight line interpolation of `dgam` has been employed to stabilise simulations, though this can be relaxed if the spatial resolution is increased. This γ related instability is also related to the choice of magnitude for the initial turbulent energy profile. If this value is chosen below 0.1`keV`, straight line interpolation must be employed for `dgam` to stabilise the scheme. The growth parameter λ is limited by numerical instabilities, namely the region in the vicinity of ρ_{ICRH} on the edge-side. A temperature profile instability builds in this region at approximately 0.5`s` for certain parameter and resolution combinations. Computation time is short for initial turbulent energy profiles in the range $10^{-3} - 10^{-2}$`keV`, while simulations are more unstable, thus increasing computation time, for initial turbulent energy profiles for 10^{-5} or 10^{-1}`keV`.

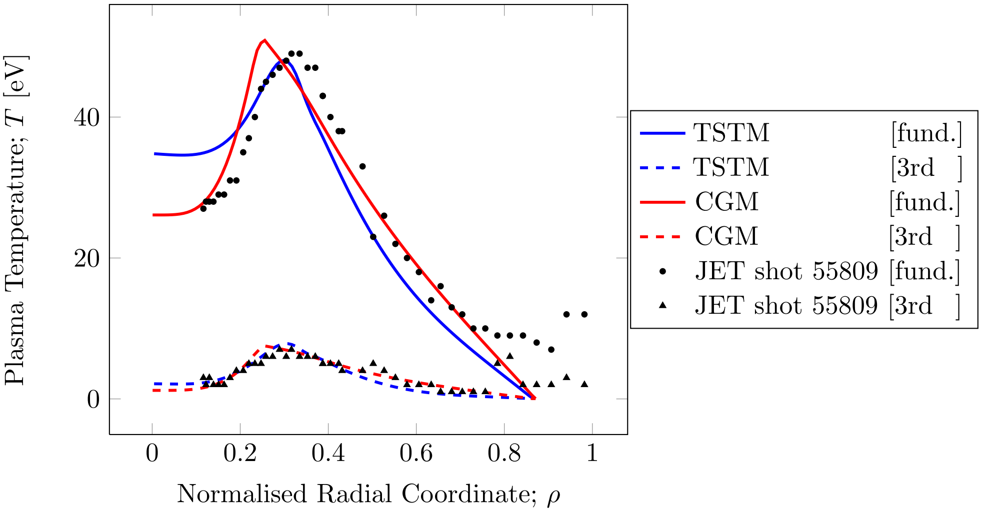
As discussed in section 3.3, if the gyroBohm scaling factor, Eq. (3.5), is included in the neo-classical diffusion term the temperature profile lacks the CGM characteristics outlined in section 3.4, based on e.g. [9], i.e. a non-stiff core region, a stiff middle region and an edge region with logarithmic temperature gradient well above threshold. Instead, the core piedestal is absorbed in a balloon-like temperature profile, see Fig. 3.2. In that case, the γ profile is only positive in the plasma edge region, $\rho \in [0.6, 0.87]$, i.e. with gyroBohm scaling on the collisional term, stiffness is confined to the edge region. Consequently, gyroBohm TSTM cold pulse results show very strong damping of the pulse in cold pulse experiments, which contradicts experimental data. Removal of the gyroBohm factor breaks this stiff region confinement, and anomalous fluxes are observed to be in the same radial ranges as for CGM experiments (compare Fig. 3.6 (a) to Figs. 3.13 and 3.15).

It is observed, that having the gyroBohm term on the the neo-classical term results in simulation failure if the initial turbulent energy profile is too great. If the gyroBohm term is not included, the simulation fails if the initial turbulent energy profile is too small.

Loosing the gyroBohm and q dependence should also make the TSTM, Eqs. (3.10-3.11), directly comparable to the CGM, Eq. (3.7). Without the gyroBohm term, a factor of approximately 3 on Eq. (3.11) has been used so far, in order to fit the TSTM steady-state profile to that of JET shot 55809. The cold pulse propagation speed is then less than half the one found with the CGM. Alternatively, the source profile, Eq. (3.1), can be divided by this fitting factor to acheive a steady-state in the vicinity of the experimental data. The source profile applied in CGM simulations would thus differ from the source profile applied in TSTM simulations, consequently leaving the models incomparable. For this reason alone modifying the source profile should be avoided. Upon numerically solving the TSTM with a reduced source profile it is found that the flat piedestal at the core (sharp boundary) has been 'eroded', i.e. the temperature profile is neither balloon-like nor does it feature a core piedestal; rather, it goes in a straight line from the inhomogeneous Dirichlet condition at the edge towards the core, bending smoothly when approaching the core to achieve the homogeneous Neumann condition at the plasma centre. As the options discussed above seem insufficient, the unavoidable road ahead is to try different combinations of TSTM parameter values. Dropping the gyroBohm scaling on the neo-classical diffusion term renders the parameters listed in [28] obsolete. The source profile used in CGM simulations is retained so CGM and TSTM results can be compared. For the same reason, $\kappa_c = 1.8$, is retained for TSTM simulations. A new set of C, λ and χ_0 values must be found. Besides these, simulation results depends strongly on the choice of turbulent energy initial profile, as well as the turbulent energy parameters D_0, γ_0 and β are fitting parameters.

In Fig. 3.12 the heat modulation results for the CGM, Eq. (3.7) and TSTM, Eqs. (3.10-3.11), are shown. Notice in Fig. 3.12 (b) the much slower heat wave propagation predicted by the TSTM, than for the CGM. The two models are, however, not compared for the same neo-classical diffusion coefficient, χ_0, as it has not been possible to obtain a stable temperature profile with the TSTM at the CGM value $\chi_0 = 0.6\mathtt{s}^{-1}$. The twice as large value (see table 3.3) has been found to stabilise the $\rho = 0$ time trace somewhat. The TSTM parameters used to obtain Fig. 3.12, are listed in table 3.3. During heat modulation, the neo-classical transport to anomalous transport ratio should be small when comparing to cold pulse experiments, due to the less steep temperature gradients induced

(a) Amplitude profile: JET shot 55809 data compared to CGM and TSTM



(b) Phase profile: JET shot 55809 data compared to CGM and TSTM

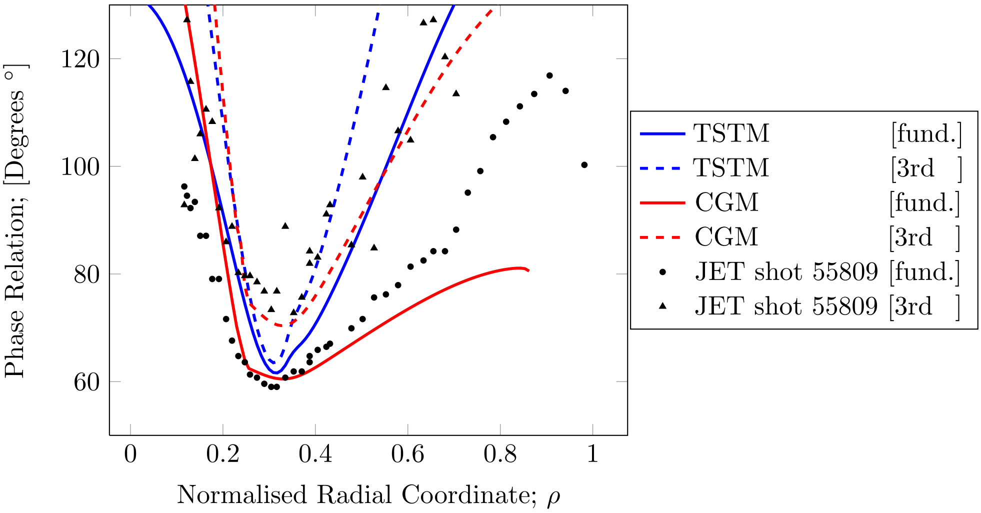


Figure 3.12: Heat modulation fundamental mode and 3rd harmonic amplitude and phase profiles predicted by the TSTM, compared to JET shot 55809 data. Fig. 3.9 CGM results are also shown. *TSTM parameters*: listed in table 3.3. *Source parameters*: $\mathcal{A}(t) = 0.75 \pm 0.5$. *Spatial resolution*: $\# \ \rho$: 100. *Temporal resolution*: $\mathsf{RelTol} = \mathsf{AbsTol} = 10^{-4}$.

Table 3.3: Set of parameters used for TSTM investigations

Parameter:	C	λ	χ_0	κ_c	D_0	γ_0	β
Value:	10.5	1.0	1.2	1.8	35.0	0.5	1.0

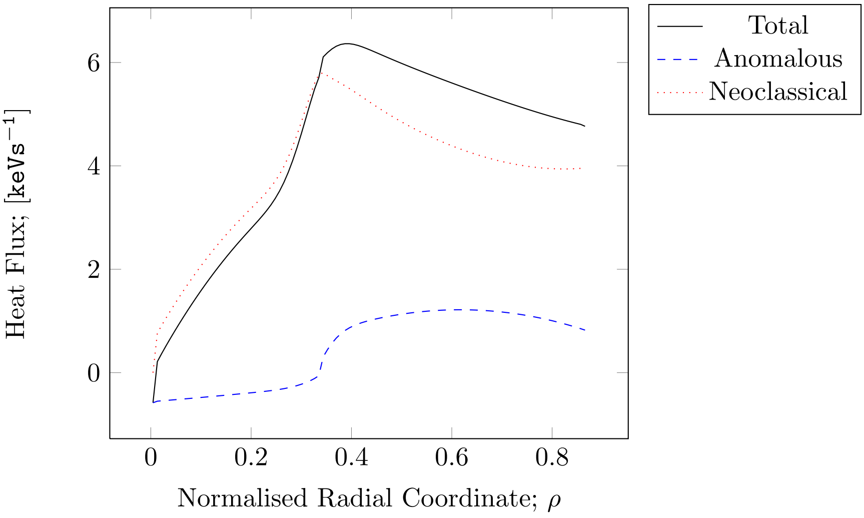
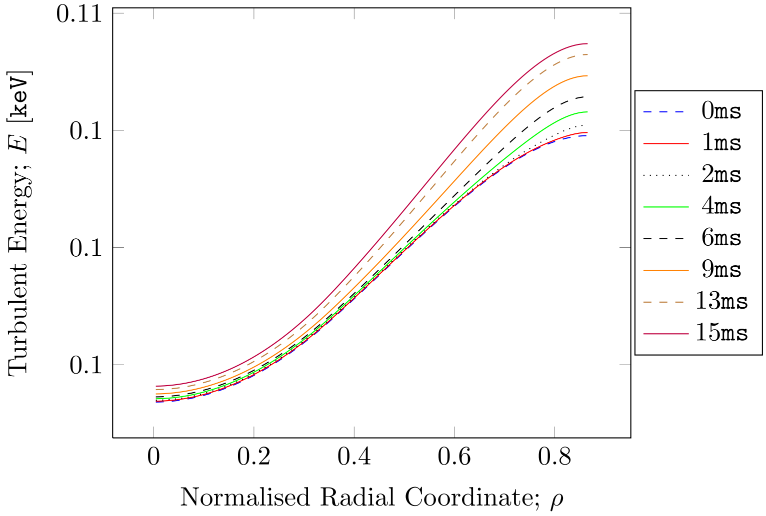


Figure 3.13: Steady-state fluxes for the TSTM parameters listed in table 3.3, for initial turbulent energy profile at 0.1`keV`. *Spatial resolution*: # ρ: 100. *Temporal resolution*: RelTol = AbsTol = 10^{-4}.

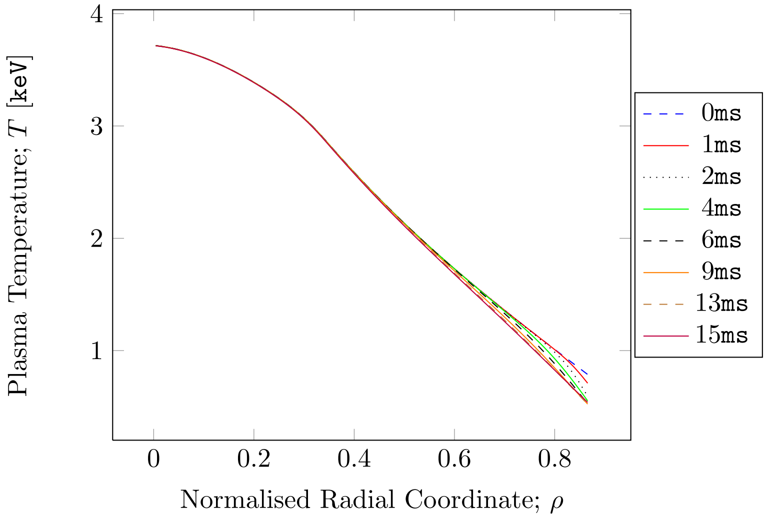
during heat modulation. Fig. 3.13 shows that the steady-state anomalous flux is small and switches sign around $\rho = \rho_{\mathrm{ICRH}}$, from subcritical in the core region (upgradient) to turbulent (downgradient) in the middle and edge regions.

The fundamental mode phase data from JET shot 55809 seems to be approximately in the middle of CGM and TSTM phase results. This might indicate that $\chi_0 = 0.9$ could be a suitable basis for future parameter fitting scans. Note, that results in Figs. 3.12 and 3.14 are obtained with $\mathcal{A}(t)$=0.75 oscillating between 0.25 and 1.25, the same as in [28].

Fig. 3.14 shows, that the turbulent energy profile rises, particularly at the edge, as the cold pulse propagates through the plasma, as also observed in [28]. This behaviour is expected from the TSTM, Eqs. (3.10-3.11), since the sudden cold



(a) Turbulent energy profile during cold pulse



(b) Plasma temperature profile during cold pulse

Figure 3.14: Turbulent energy and plasma temperature radial profiles during cold pulse, for TSTM parameters listed in table 3.3. *Spatial resolution*: # ρ: 100. *Temporal resolution*: RelTol = AbsTol = 10^{-4}.

pulse drop at the plasma edge induces strong growth in the $\frac{\partial E}{\partial t}$ RHS via γ which depends on $\frac{\partial T}{\partial \rho}$. However, the profile is $\sim$ an order of magnitude smaller, the change during cold pulse $\sim$ two orders of magnitude smaller. So it seems the coupling of turbulent energy to the temperature dynamics is weak, compared to the case in [28], which is substantiated by the low steady-state anomalous flux observed in Fig. 3.13. The task is to find a set of parameters for which Fig. 3.14 (b) is perturbed strongly, all the way to the core.

In the TSTM, as presented in [28], the anomalous flux term has no explicit ∇T dependence, as is the case for the CGM anomalous flux in Eq. (3.7). No such dependence emerges from the motivation for the TSTM anomalous flux, described in section 1.7. However, assuming such dependence in this semi-heuristic model could be argued for: Assume Eq. (1.24) forms the physics basis of the thermal dynamics in a tokamak. One can then argue pro including ∇T proportionality in the anomalous flux term, as for the neo-classical term. In this section, the impact of implementing this modification in Eq. (3.11) is tried out, though it is stressed that such a test is unrooted in any strong physics argument. Eq. (3.11) then becomes

$$\frac{\partial T}{\partial t} = -\nabla \cdot CET \tanh(\gamma)\nabla T + \chi_0 \nabla T. \tag{3.14}$$

This change is found not to affect the steady-state profile notably. However, the modification introduces an interesting feature in cold pulse numerical results. Strong polarity reversal is observed at the plasma core ($\rho = 0.11$) for parameters listed in table 3.1, i.e. the cold pulse at the plasma edge, $\rho = 0.87$, induces a temperature rise at the core, $\rho = 0.11$, on a very short time-scale, before the cold pulse itself reaches the core. This phenomenon was first observed experimentally in the TEXT device [11]. The CGM, which do not feature upgradient transport, does not reproduce this feature. Polarity reversal is also observed without the added ∇T factor on the anomalous flux term, though smaller in magnitude. In [28] polarity reversal is accounted for via the turbulent energy profile adjusting on a very short time-scale. If the anomalous flux constant, C, is decreased from 0.5 to 0.1, the polarity reversal at the core almost disappears. For $C \gg 0.5$ the pulse propagation time increases. The polarity reversal numerical experiments are conducted with an initial turbulent energy profile at $= 0.01$`keV`, and a factor of 3 on the Eq. (3.11) RHS.

It is found, that cold pulse propagation speed does not change significantly when increasing the spatial resolution. # ρ: 100 seems to be sufficient to resolve the experiments spatially. Variations in $\mathcal{A}(t)$ do not produce notable changes in propagation speed either. As is the case for the CGM implementation (see Fig. 3.11), switching from central approximation to 5-point stencil does not change propagation speeds, though using the $3rd$ order accurate scheme do produce increased edge-to-core propagation speeds.

Keeping χ_0 fixed at $0.6\mathtt{s}^{-1}$ no set of TSTM parameters could be found for which propagation to the core occured faster than described by the CGM. The edge pulse impact on the core observed, as modeled by the TSTM, is limited, when comparing to the CGM, i.e. the pulse-induced drop in temperature at the core is smaller. However, clear polarity reversal at the core is observed, without the assumption made in Eq. (3.14), at specific parameters: $C = 5.5$, $\chi_0 = 1.6$, $\lambda = 1.7$, remaining parameters as in table 3.3, and initial turbulent energy profile at $0.001\mathtt{keV}$. Varying e.g. λ or the initial turbulent energy profile slightly, makes the polarity reversal disappear, i.e. the implementation is quite unstable.

In general, TSTM modeled cold pulse and modulation dynamics is extremely sensitive to changes in all parameters. The apparent temperature steady-states observed for low initial turbulent energy profiles ($\sim 10^{-3}\mathtt{keV}$), only appear converged since the anomalous flux term is small, as observed in Fig. 3.13. Time trace analysis reveals, that the turbulent energy profile keeps changing for almost half a minute, before the profile actually converges to a steady-state. This qualitative behaviour is also observed in section 3.3. The turbulent energy profile typically converges to a steady-state profile in the range $\in [0.1, 1\mathtt{keV}]$ for initial profiles in the range $\in [0.01, 10\mathtt{keV}]$. A scan of the C to λ ratio indicates, that as $\frac{C}{\lambda} \to \infty$ instability growth appears in the plasma centre time trace.

The only way found to reach core times shorter than those found for the CGM involves departure from the $\chi_0 = 0.6\mathtt{s}^{-1}$ value, which ideally should be kept common for CGM and TSTM simulations in order for results to be comparable. With $C = 11$ and $\chi_0 = 0.15\mathtt{s}^{-1}$ faster propagation is obtained and the great cold pulse temperature drop in the core looks closer to the CGM cold pulse results than for other parameters. Increasing the $\frac{C}{\chi_0}$ ratio, increases both the propagation speed and the magnitude of the cold pulse drop in the core.

For $\chi_0 = 0.11\mathtt{s}^{-1}$ and $C = 11$ – remaining TSTM parameters as in table 3.1 – a very fast edge-to-core cold pulse propagation is observed, using the central approximation stencil. The time delay for the cold pulse to reach the core is $\sim 10\mathtt{ms}$. Analysing the radial profile for the time delay shows that propagation in the plasma edge region is slightly slower than for the CGM results, Fig. 3.11. However, in the stiff middle region the propagation speed becomes very fast due to the wide stiffness region for which the logarithmic temperature gradient is very close to the critical gradient, κ_c. The steady-state heat fluxes for these parameters are shown in Fig. 3.15. Observe, that the anomalous flux now dominates. As for the CGM in sections 3.5.1-3.5.4, the strange kink at the sub- to super-critical transition (around $\rho \approx 0.15$) in Fig. 3.15) is also present in this TSTM implementation. This suggests that the spatial region in which the transition from sub- to super critical transport occurs may be underresolved. However, simulation wall-time rises rapidly when spatial as well as temporal

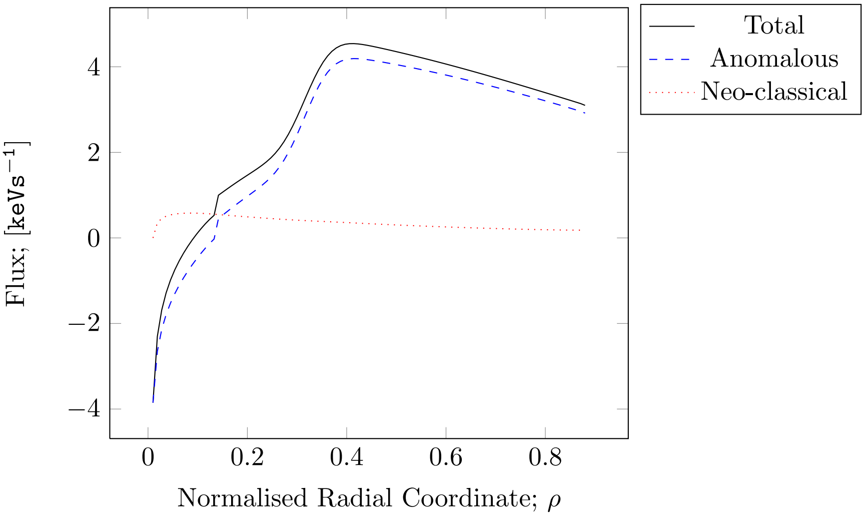


Figure 3.15: Steady-state fluxes for initial turbulence energy 0.1`keV`. *TSTM parameters*: $C = 11$, $\chi_0 = 0.11\mathtt{s}^{-1}$, remaining parameters as in table 3.1. # ρ: 100. *Temporal resolution*: $\mathsf{RelTol} = \mathsf{AbsTol} = 10^{-5}$.

resolution is increased, slowing down investigations.

Results presented in section 3.3 demonstrates, that steady-state for the radial plasma temperature is not achieved for the implementation of the TSTM defined in [28]. In this section, the effect of dismissing the gyroBohm factor on the neoclassical transport has been investigated. The modeling results are found to be very sensitive to the choice of initial condition chosen for the turbulent energy profile, as well as the other TSTM parameters, and again the TSTM implementation does not display true steady-state convergence within the time frame dictated by experiment (∼ 100`ms`). The CGM results presented in Figs. 3.6, 3.7 and 3.8 (b) indicate, that the explicit derivatives scheme outlined in section 2.2.2 may not be a good approach. Fig. 3.6 (b) indicates that the temperature profile in the spatial middle and edge regions oscillates around an equilibrium formed with the accumulated source, due to the anomalous heat flux. This oscillation is not visible in the plasma centre time traces as that is in the sub-critical core region. Motivated by these inferior results, a new implementation is developed, focusing on improving results and reducing computation time such that poloidal plane simulations can be run within a reasonable time frame.

3.6 Direct Derivatives Approach Investigations

Due to the shortcomings of the results presented in section 3.5, a new numerical implementation is developed. This alternative approach is discussed in section 2.2.3, and the main features of both implementations are listed in table 3.4. ↑ indicates and advantage, whereas ↓ indicates a disadvantage, when comparing the explicit- and direct derivative approaches. ↺ indicates neither advantage nor disadvantage. Based on the sensitivity on parameters of the TSTM implementation used to produce results in section 3.5.5, it is assessed that there is insufficient time to attempt to get a new TSTM implementation working. In this section, and for the remainder of this chapter, the focus will therefore be on the CGM, Eq. (1.25).

Table 3.4: Key differences between old and new implementations

`OLD IMPLEMENTATION`	`NEW IMPLEMENTATION`
↑ Spatial stencil flexibility	Hardcoded central approximation ↓
↺ Explicit LHS quantity derivatives	Direct derivatives of operands ↺
↺ BCs for LHS quantities only	BCs required for each operand ↺
↓ Several arithmetic operations	Reuse of calculated quantities ↑
↓ External function calls each step	Everything in `ode23tb` argument ↑

The safety factor, q, is assumed radially dependent and is shown in table 3.5, along with the other parameters used in [28]. Ghost points are included in the relevant quantity arrays, and the arrayspan denoting the physical grid points, i.e. the full dimension excluding ghost points, is denoted `no_g`. Quantities are defined and used for further calculation, i.e.

```
qT32 = T(no_g).*sqrt(T(no_g));
q32mat = repmat(q32,1,1);
qT32 = q32mat.*qT32;
chi_collisional = chi0*qT32;
```

where `q32` is the safety factor profile raised to the $\frac{3}{2}th$ power. The logarithmic temperature gradient and γ functions are evaluated as

```
log_grad(no_g) = inv2dr*log(T(3:N)./T(1:N-2));
```

```
gamma          = lambda*max(-log_grad(no_g)-crit_grad,0.0); % Heaviside
gam            = sqrt(gamma);
posgrowth      = logical(imag(gam));
gam(posgrowth) = -sqrt(-gamma(posgrowth));
```

and thereafter the heat fluxes,

```
chi(no_g) = C*gam.*qT32;
anom_flux = chi(no_g).*T(no_g).*log_grad(no_g);
neoc_flux = chi_collisional.*T(no_g).*log_grad(no_g);
total_flux = anom_flux + neoc_flux;
chi(no_g) = chi(no_g) + chi_collisional; % total chi
```

The additional boundary conditions needed, are chosen as

```
chi(1) = chi(2); chi(N) = chi(N-1); % chi has homogeneous Neumann at boundaries
log_grad(1) = log_grad(2);     % log_grad has homogeneous Neumann at r = 0
log_grad(N) = 2*log_grad(N-1) - log_grad(N-2); % slope continues at r = 0.87
```

and with this, the CGM RHS is written

```
invrmat = repmat(1./r,1,1);
dT_dt = FH*(invrmat.*T(no_g).*chi(no_g).*log_grad(no_g) ...
           + inv2dr*(T(3:N)-T(1:N-2)).*chi(no_g).*log_grad(no_g) ...
           + T(no_g).*inv2dr.*(chi(3:N)-chi(1:N-2)).*log_grad(no_g) ...
           + T(no_g).*chi(no_g).*inv2dr.*(log_grad(3:N)-log_grad(1:N-2)));

S = NBI+amp*RF;

% add source
span = 1:length(r)-1; % no source on edge (no material to heat)
                      % ((and undermines T edge BC definition))

dT_dt(span) = dT_dt(span)+S(span);
```

Going from the explicit derivatives- to direct derivatives scheme, one tradeoff is the loss of ability to control spatial accuracy via the spatial discretisation chosen. Spatial accuracy is thus determined solely by the ρ resolution. However, conducting derivative operations on numerically evaluated quantities directly limits the risk of error, as the minimum of arithmetic operations is required. Furthermore, if there are good physics based arguments for choices of additional boundary conditions needed, the addition of these allows for a closer relation between the numerical implementation and the plasma conditions in the tokamak. Then, one has better control over the implementation to imitate the actual physics. Finally, the new code runs much faster than the old, due to the elimination of external function calls in the new code.

Table 3.5: Parameters used in [28], and q-profile used in the C code [29]

Parameter:	C	λ	χ_0	$q(\rho)$
Value:	0.5	1.7	0.6	$1+\rho+6\rho^4$

The code used to generate results presented in this section is found in appendix C

3.6.1 Steady-State

The temperature profile rapidly undergoes a significant rise in magnitude prior to reaching steady-state. One might infer that changing the initial temperature profile from flat (with edge Dirichlet value) to a profile closer to that of the steady-state should reduce computation time. Consequently, the numerical estimate of the time it takes the profile to converge to steady-state (evident from the temperature time trace of $\rho = 0$) becomes 'less physical', as a flat initial temperature profile better resembles the physical conditions before heating. Other than that, steady-state-, heat modulation- and cold pulse results should be unaffected by this change in the temperature initial condition, and should in theory be obtained quicker. The only physical quantity estimate lost by doing this would be the numerical estimate of the time it takes to heat the profile to steady-state. Denoting the steady-state temperature measured at the plasma edge, $\rho_{\text{edge}} = 0.87$, in JET shot 55809 by T_{edge}, the initial temperature profile, $T_{\text{IC}}(\rho)$, is taken to be

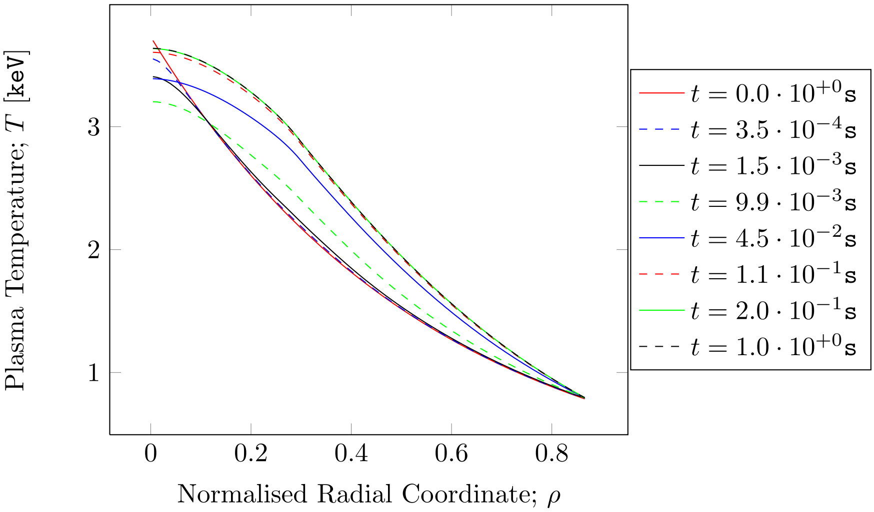
$$T_{\text{IC}}(\rho) = T_{\text{edge}} \exp\left(\kappa_c[\rho_{\text{edge}} - \rho]\right). \tag{3.15}$$

The time it takes the system to reach steady-state is approximately 100`ms`. As Fig. 3.16 shows, the $\rho = 0$ time trace first dives before increasing and converging to the steady-state.

At this point, it is considered instructive to visualise a key difference between the CGM and TSTM, namely that of upgradient anomalous transport only featured in the TSTM. Therefore, steady-state results for the CGM are here presented both with- and without the inclusion of small upgradient anomalous transport.

Fig. 3.17 illustrates the difference between adding small upgradient transport in the sub-critical temperature profile region – as featured in the TSTM – and having the Heaviside function, which controls the anomalous flux term in all CGM

(a) Steady-state convergence: Temperature profiles for selected times t



(b) Steady-state stiff region

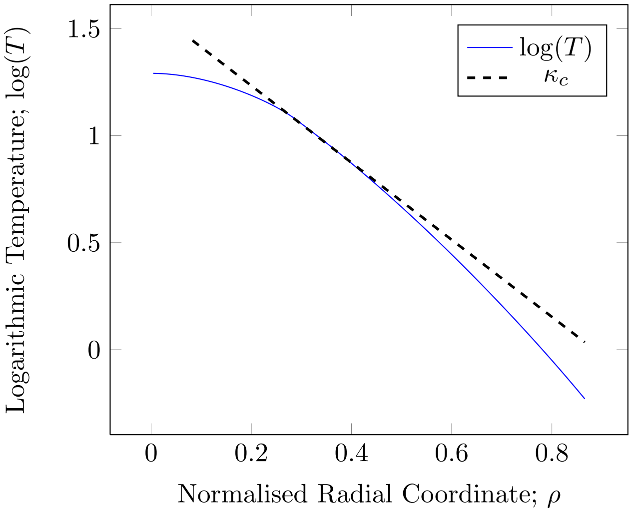
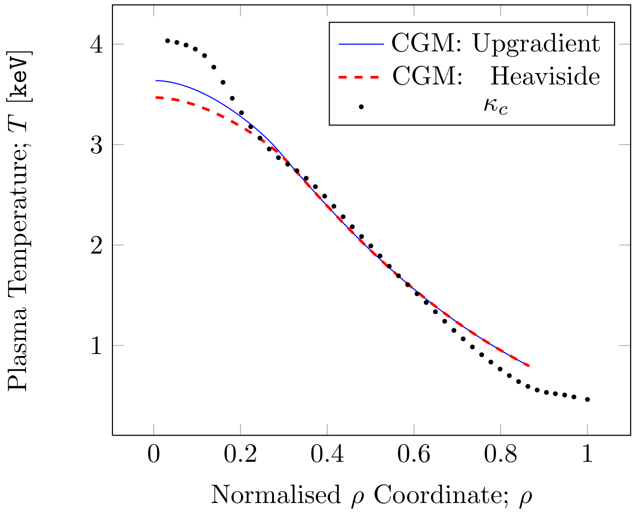


Figure 3.16: (a): Steady-state convergence for the temperature profile predicted by the CGM, Eq. (1.25). (b): Visualisation of the spatial extend in which the logarithmic gradient is in the vicinity of the critical gradient threshold, κ_c (the stiff region). The CGM parameters used are shown in table 3.5. *Spatial resolution*: # ρ: 100. *Temporal resolution*: $\mathsf{RelTol} = \mathsf{AbsTol} = 10^{-4}$.

(a) Steady-state temperature profiles



(b) Deviation between LHS and RHS of Eq. (3.16)

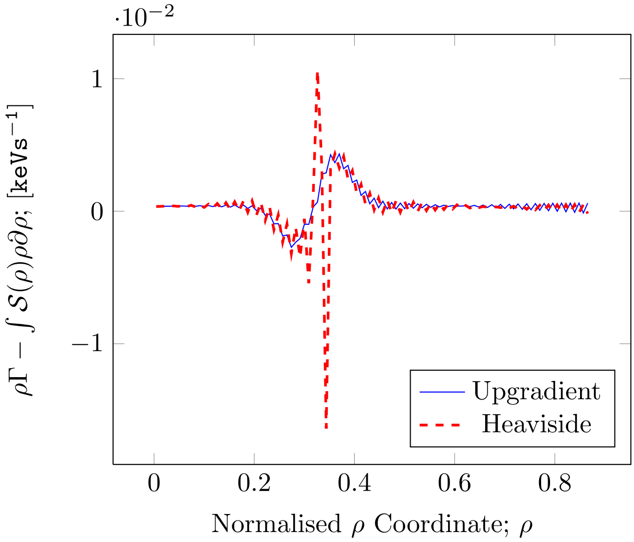
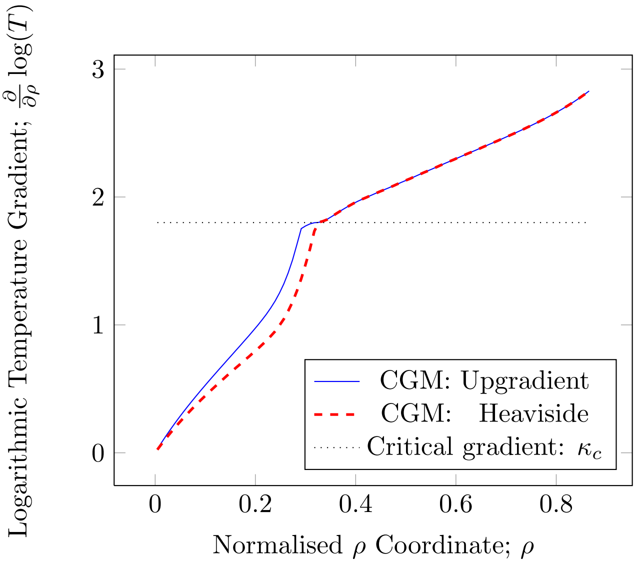


Figure 3.17: (a): Steady-state for the temperature profile predicted by the CGM, Eq. (1.25), for anomalous heat flux Heaviside and small upgradient factors, respectively. (b): Deviation between LHS and RHS in Eq. (3.16). The CGM parameters used are shown in table 3.5. *Spatial resolution*: # ρ: 100. *Temporal resolution*: $\mathsf{RelTol} = \mathsf{AbsTol} = 10^{-4}$.

(a) Logarithmic temperature gradient profile



(b) Heat fluxes for the Heaviside and small upgradient cases

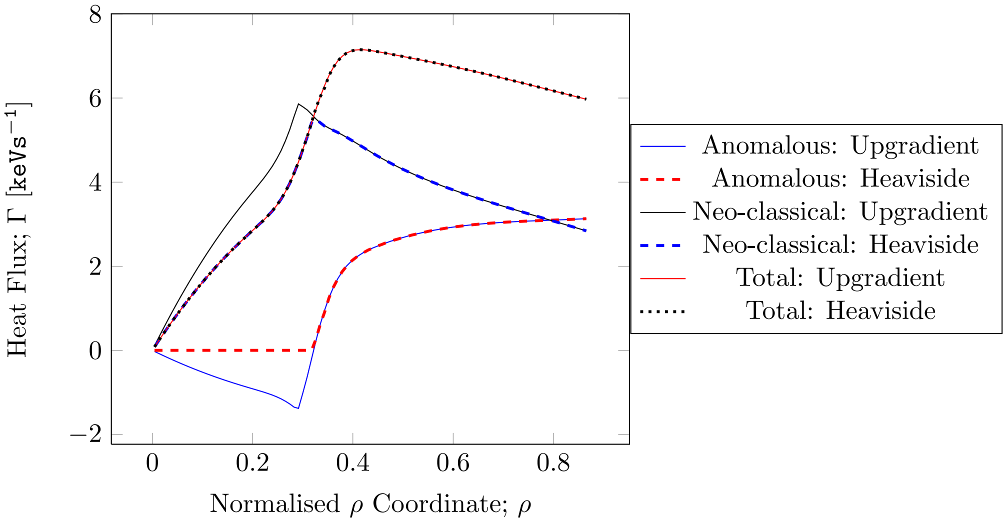


Figure 3.18: Steady-state logarithmic temperature gradients (a) and heat flux profiles predicted by the CGM, Eq. (1.25), for anomalous heat flux Heaviside and small upgradient factors, respectively. CGM parameters are shown in table 3.5. *Spatial resolution*: # ρ: 100. *Temporal resolution*: RelTol = AbsTol = 10^{-4}.

definitions. Notice the higher core temperature for the case where upgradient transport is included. This observation agrees with the expected behaviour, as the upgradient transport kicks in at the core where the logarithmic temperature gradient is below the threshold, κ_c. Analogous to Eq. (3.13), the relation

$$\int \mathcal{S}(\rho)\rho\partial\rho = -\rho\Gamma \tag{3.16}$$

should be satisfied in steady-state. As Fig. 3.17 (b) indicates, the Heaviside case seems more unstable than the case including small upgradient transport. However, both these direct derivatives implementation cases agree much better with Eq. (3.16) than the explicit derivatives implementation do with Eq. (3.13) in section 3.5.1. Fig. 3.16 (a) shows that this region of instability is in the vicinity of the steepest temperature gradients on the ρ interval, and Fig. 3.16 (b) demonstrates that this region is the stiff region. As might be expected from Fig. 3.17 (b), there is a notable difference in computation time between the two cases. The simulation of the Heaviside case takes more than 30 times longer (≈ 135s) than the upgradient case (≈ 4s).

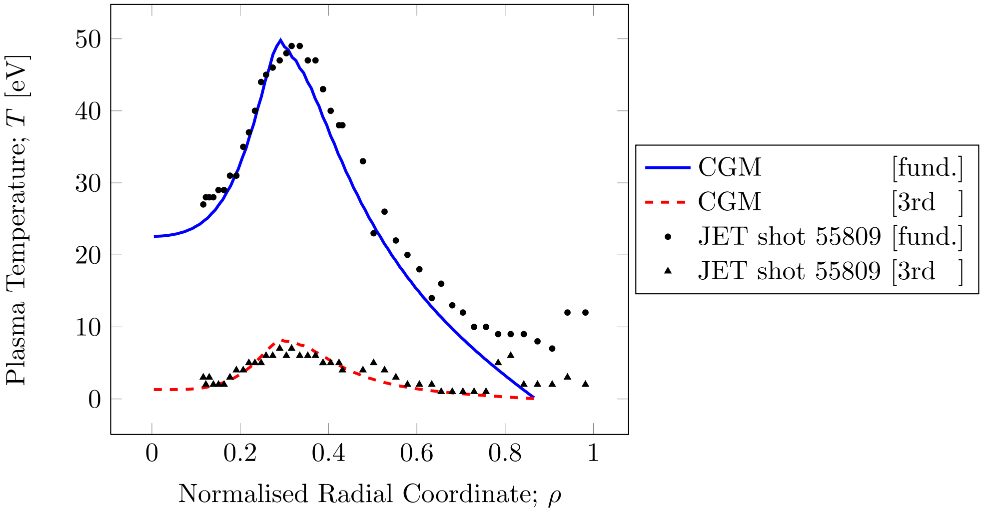
The logarithmic temperature gradients corresponding to the steady-state profiles shown in Fig. 3.17 (a) are plotted in Fig. 3.18 (a). Fig. 3.18 (a) demonstrates, that turbulence starts to develop closer to the core region for the upgradient case, though the downgradient anomalous flux is invoked $\sim \Delta\rho = 0.05$ further towards the plasma edge for both cases. This is demonstrated more directly in Fig. 3.18 (b). Observe in Fig. 3.18 (b), that the total heat fluxes for the Heaviside and upgradient cases are very close, while the anomalous- and neo-classical fluxes differ. Notice for the upgradient case how the neo-classical heat flux compensates for the upgradient transport in the core region.

In this section, upgradient anomalous transport has been included in the CGM in order to test the implementation, even though it is not a feature of the model. For the remainder of this chapter only anomalous transport controlled by the Heaviside function is considered.

3.6.2 Heat Modulation Amplitude & Phase Profiles

The ICRH heat modulation results shown in Fig. 3.19 are obtained with the CGM, Eq. (1.25). The amplitude results agrees fairly well with experiment, except in the edge region, due to the inhomogeneous Dirichlet condition on the temperature forced there. Contrary to the modulation results for the explicit derivatives implementation of the CGM, Eq. (3.7), (see section 3.5.3) Fig. 3.19 (b) shows that the heat wave propagation speed is generally too slow, though a

(a) Amplitude profile: JET shot 55809 data compared to CGM



(b) Phase profile: JET shot 55809 data compared to CGM

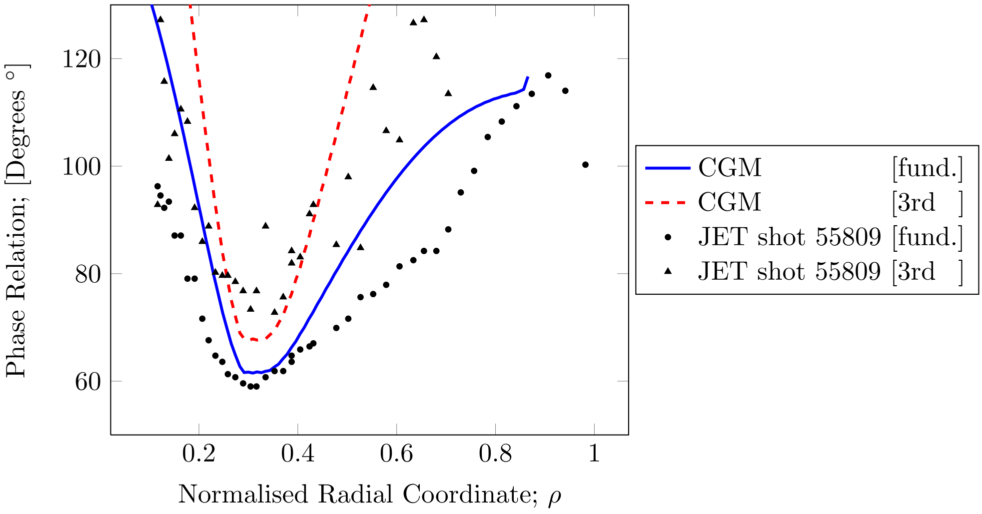


Figure 3.19: Heat modulation fundamental mode and 3rd harmonic amplitude and phase profiles predicted by the CGM, Eq. (1.25), compared to JET shot 55809 data. *Source parameters*: $\mathcal{A}(t) = 0.75 \pm 0.55$. *Spatial resolution*: $\#\ \rho$: 100. *Temporal resolution*: $\mathsf{RelTol} = \mathsf{AbsTol} = 10^{-5}$.

speed increase is observed for the fundamental mode going towards the plasma edge.

3.6.3 Cold Pulse Impact on Steady-state

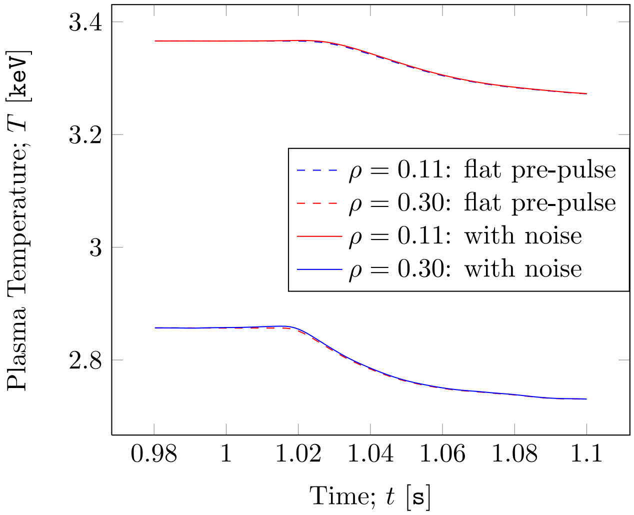
Initially, polarity reversal has been observed in the core; i.e. the occurance of a slight temperature increase before the actual cold pulse drop reaches the core. However, polarity reversal should not occur as the CGM features no upgradient transport. With regard to the phenomenon of polarity reversal, two points must be checked:

1. That steady-state is reached.
2. That the cold pulse edge data is smooth before the cold pulse.

As already tested in section 3.6.1, the time trace of the CGM data shows that the temperature profile reaches steady-state after $\sim$ 100`ms`. Inspection of the JET shot 55809 cold pulse data revealed the true origin of this pseudo-phenomena. A noise peak is observed in the cold pulse edge data shortly before the cold pulse is triggered and removal of this noise from the applied cold pulse edge Dirichlet data also removes the false polarity reversal in the core. The pre-pulse noise in the experiment edge data gives rise to a temperature increase in the core of the order of `1eV`, just before the cold pulse drop (of reverse sign and much greater magnitude) arrives at the core. The cold pulse data – with and without noise – is shown in Fig. 3.20. Fig. 3.21 (a) displays slightly longer edge-to-core CGM cold pulse propagation time than the 22`ms` in [28] at $\rho = 0.11$. Notice the very fast propagation in the $\rho \in [0.2, 0.4]$ region. As Fig. 3.16 (b) shows, this is the stiff region. Fig. 3.21 (a) shows that fast propagation occurs in the super-critical edge and stiff middle regions, with fast pulse propagation from edge to the sub-critical core region in which the pulse slows. In Fig. 3.21 (b) the effect of the cold pulse perturbation at the plasma edge is visualised.

The implementation used in this section is now extended in the poloidal dimension. This allows for investigations of poloidal effects on the CGM modeling results.

(a) Cold pulse time traces for $\rho = 0.11$ and $\rho = 0.30$



(b) Cold pulse time traces for $\rho = 0.60$ and $\rho = 0.87$

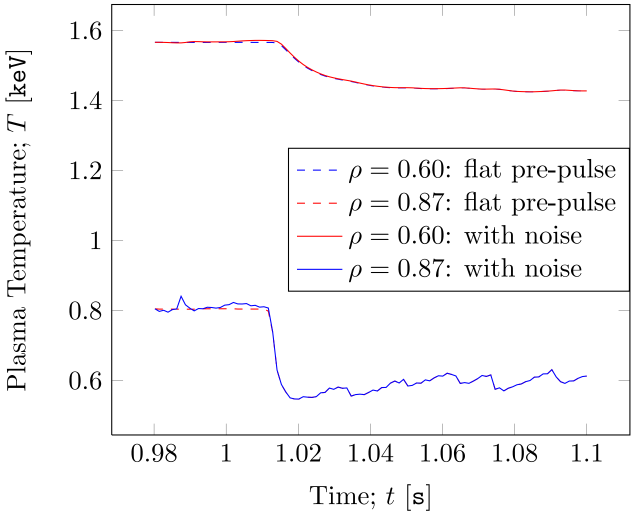
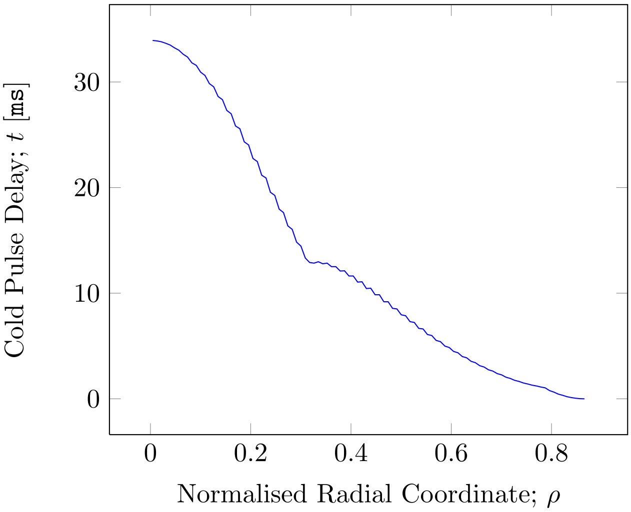


Figure 3.20: Cold pulse time traces predicted by the CGM, Eq. (1.25). CGM parameters are shown in table 3.5. *Spatial resolution*: # ρ: 100. *Temporal resolution*: RelTol = AbsTol = 10^{-5}.

(a) Cold pulse 30`eV` drop time delay



(b) Plasma temperature profile during cold pulse

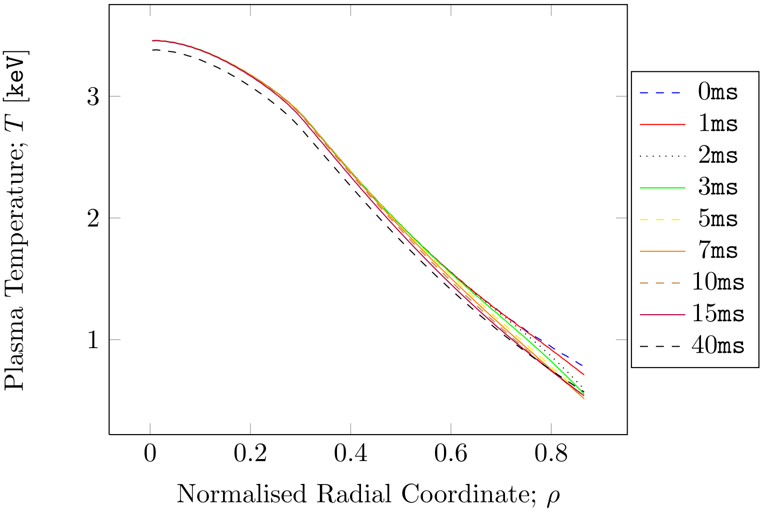


Figure 3.21: Time delay from edge to core (a) and temperature profile during cold pulse (b), modeled by the CGM, Eq. (1.25), with parameters as shown in table 3.5. *Spatial resolution*: # ρ: 100. *Temporal resolution*: RelTol = AbsTol = 10^{-5}.

3.7 Poloidal Cross-section Modeling

For the extension to 2D, a logical first step is to keep the diffusion coefficients and the radial source profile, expressed by Eqs. (3.1-3.3), and extend the model in the poloidal dimension. Then CGM steady-state, heat modulation and cold pulse results for any poloidal angle φ must match the 1D results presented in section 3.6. Next step is to switch from symmetric to outboard ICRH heating in order to model actual reactor conditions numerically. Finally, the poloidal component of the fast dynamics along the magnetic field lines is included in the CGM in order to model the 2D fusion plasma, and 2D results are then compared to 1D results.

The developed MATLAB codes producing the results presented in this section are available in appendix D.

3.7.1 Boundary Conditions

For the radial part, $\rho = 0$ is no longer a physical limit of the spatial extend of the plasma, only the temperature value measured at the plasma edge during JET shot 55809 is retained as an inhomogeneous Dirichlet condition there. Thus, the homogeneous Neumann condition at $\rho = 0$ is no longer required. Radial $\rho = 0$ ghost points are instead set equal to the physical ρ centre value a poloidal distance $\Delta\varphi = \pi$ away. This restricts the allowed poloidal resolutions used in simulations to even numbers. For the poloidal part, the only physically sound boundary condition is to require 2π periodicity for all quantities in the poloidal direction.

Quantity arrays for which boundary conditions are applied via ghost points include:

- The plasma temperature array, `T`,
- the logarithmic temperature gradient in the radial dimension, `log_grad_r`,
- the logarithmic temperature gradient in the poloidal dimension, `log_grad_p`,
- the radially dependent diffusion coefficient, `chi_r`.

The structure of the 2D array input to the `ode23tb` time-integrator is shown in table 3.6.

Table 3.6: Input array for `ode23tb` in the 2D implementation

`Array row #`	ρ `grid point`	φ `grid point`
1	$\frac{\delta\rho}{2}$	0
$\vdots$	$\vdots$	0
N_ρ	$\frac{[N_\rho-1]\delta\rho}{2}$	0
$N_\rho + 1$	$\frac{\delta\rho}{2}$	$\delta\varphi$
$\vdots$	$\vdots$	$\delta\varphi$
$2N_\rho$	$\frac{[2N_\rho-1]\delta\rho}{2}$	$\delta\varphi$
$\vdots$	$\vdots$	$\vdots$
$N_\rho \times N_\varphi - N_\rho + 1$	$\frac{\delta\rho}{2}$	$N_\varphi\delta\varphi$
$\vdots$	$\vdots$	$N_\varphi\delta\varphi$
$N_\rho \times N_\varphi$	$\frac{[2N_\rho-1]\delta\rho}{2}$	$N_\varphi\delta\varphi$

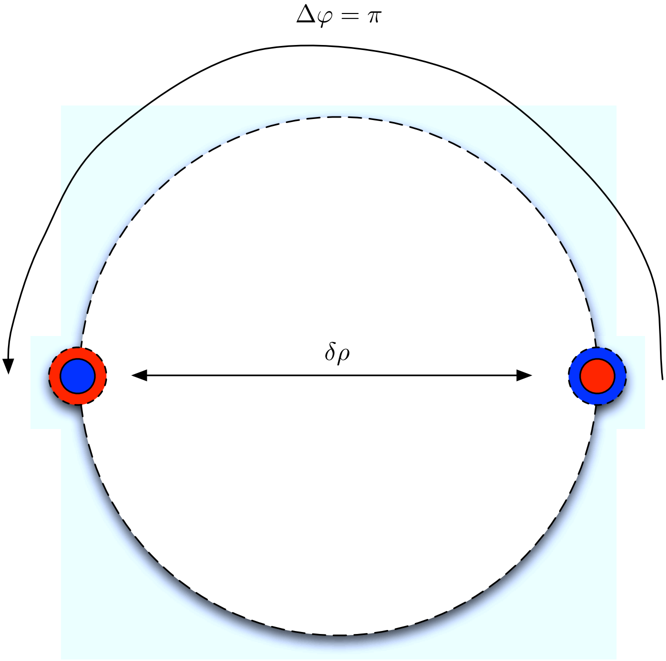


Figure 3.22: The physical grid points at the plasma centre are sketched for $\varphi = 0$ (red) and $\varphi = \pi$ (blue), with their respective radial ghost points depicted as annuli, a poloidal distance $\Delta\varphi = \pi$ away.

3.7.2 Symmetric Source Modeling

In this section, the implementation presented in section 3.6 is modeled with the same parameters and radial source profile, i.e. the parameters given in table 3.5, and the ρ-dependent source expressed by Eqs. (3.1-3.3). The only change is the inclusion of the poloidal dimension in the CGM. This is done by letting 1D implementation quantity arrays, with ρ-dependence in the row dimension, extend in the column dimension to include φ-dependence. The 2D CGM implemented reads [29]

$$\frac{\partial T}{\partial t} = \nabla \cdot q^{\frac{3}{2}} T^{\frac{5}{2}} \begin{bmatrix} \chi_s \Big[- R \frac{\partial}{\partial \rho} \log(T) - \kappa_c \Big] \mathsf{H}\Big(- R \frac{\partial}{\partial \rho} \log(T) - \kappa_c \Big) + \chi_0 \\ \chi_s \Big[- R \frac{\partial}{\partial \rho} \log(T) - \kappa_c \Big] \mathsf{H}\Big(- R \frac{\partial}{\partial \rho} \log(T) - \kappa_c \Big) + \chi_0 \end{bmatrix} \times \begin{bmatrix} \frac{\partial}{\partial \rho} \log(T) \\ \frac{1}{\rho} \frac{\partial}{\partial \varphi} \log(T) \end{bmatrix} + \mathcal{S} \tag{3.17}$$

With this setup, the 2D modeling results must meet the following requirements:

1. The radial temperature profile must match the 1D case for all φ,
2. the logarithmic temperature gradient in the radial dimension must match the 1D case for all φ,
3. the radial fluxes must match the 1D case for all φ,
4. the logarithmic temperature gradient in the poloidal dimension must be 0 for all φ,
5. poloidal fluxes must vanish for all φ,
6. heat modulation results must match the 1D case for all φ,
7. cold pulse results must match the 1D case for all φ.

Before performing these 7 tests involving the time-solver, the 2D code is tested against the 1D code simply by comparing quantity arrays for the very first time step, commenting out the poloidal terms in the CGM RHS in the 2D code. The ode23tb 1D and 2D argument functions models1D and models2D are forced to abort operation at the last line, after the global variables tested (see table 3.7) have been updated. This is equivalent to letting the solver run to a certain end time and testing the difference in the output when inserting the solution in

Table 3.7: Deviation between quantity arrays in the 1D- & 2D implementations

`1D Code`	`2D Code`	`Deviation`
T	T	0
qT32	qT32	0
log_grad	log_grad_r	0
anom_flux	anom_flux_r	0
neoc_flux	neoc_flux_r	0
chi	chi_r	0
dT_dt	dT_dt	0

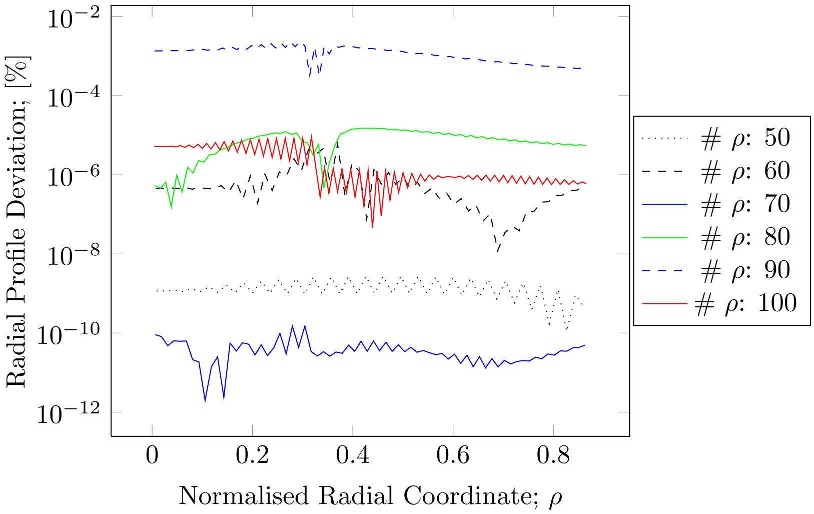
models1D and models2D, except here the solution is the initial condition profile rather than the profile at some end time. As table 3.7 shows, the 1D and 2D RHSs produce the exact same results.

Results concerning the deviation in temperature profile between the 1D and 2D implementations are shown in Figs. 3.23-3.24. From Fig. 3.23 (a) it is clear, that the deviation between the steady-state temperature profiles obtained from the 1D and 2D implementations is not 0 for any spatial resolution, which is at variance with expected results when the poloidal terms are commented out in the 2D code. In fact, the deviation varies several orders of magnitude. Note, that for # ρ: 70, the deviation between the 1D and 2D implementations is within the same order of magnitude as the inter-φ deviation ($\sim 10^{-12}\%$) between radial profiles generated by the 2D code. Fig. 3.23 (b) shows the effect of including the poloidal terms in the CGM RHS. A resolution of 70 grid points in the radial dimension is chosen for the φ scan, as Fig. 3.23 (a) shows that the deviation from the 1D result is small at this resolution. Thus, the deviation originating from the inclusion of φ RHS CGM terms is as clear as possible for this choice of radial resolution. Fig. 3.23 (b) shows that the deviation between the 1D and 2D implementations is approximately an order of magnitude higher for # φ: 20 & 40, compared to # φ: 10 & 30, in the core region, while this relation seems to invert towards the edge region. Neither Fig. 3.23 (a) nor (b) display a drop in deviation as a function of spatial resolution. This indicates that lack of accuracy of the spatial stencil is not the origin of the deviation between 1D and 2D results.

In Fig. 3.24 (a) the radial profile of the deviations in radial dimension logarithmic temperature gradient and total heat flux for 1D and 2D implementations are shown. As for the deviation in temperature profile, these should be 0. Notice the peak region in the vicinity of $\rho = 0.32$ for the deviation in total radial heat flux. This is where the ICRH heating is localised, adding a massive contribution to the NBI heating profile in the CGM source term (see Fig. 3.1). Fig. 3.24 (b) shows that the logarithmic temperature gradient and total heat flux in the poloidal dimension are well within the range of machine errors. This substantiates the 2D implementation as these quantities are expected to be 0; the heating is poloidally symmetric → no poloidal variation for any quantity.

Figs. 3.23-3.24 display finite deviations between 1D and 2D implementations, contrary to expectations. As demonstrated in table 3.7, the CGM RHS `ode23tb` input arrays return exactly the same quantity values for the 1D and 2D implementations. To ensure that the radial quantities of the 1D and 2D implementations are absolutely equivalent, the poloidal dimension is removed from all arrays in the 2D implementation. With this change to the 2D code, the exact same results for the temperature profile, logarithmic temperature gradient, fluxes, heat modulation and cold pulse delay times are obtained.

(a) Excluding poloidal RHS terms in the CGM; # φ: 10



(b) Including poloidal RHS terms in the CGM; # ρ: 70

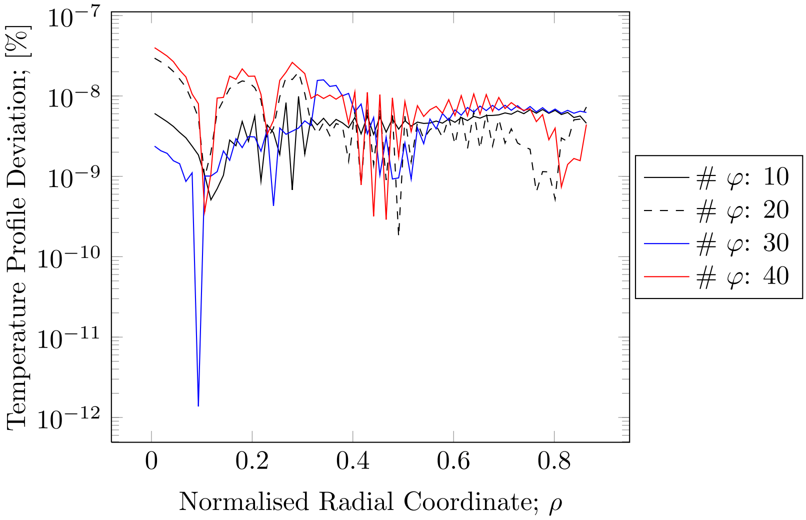
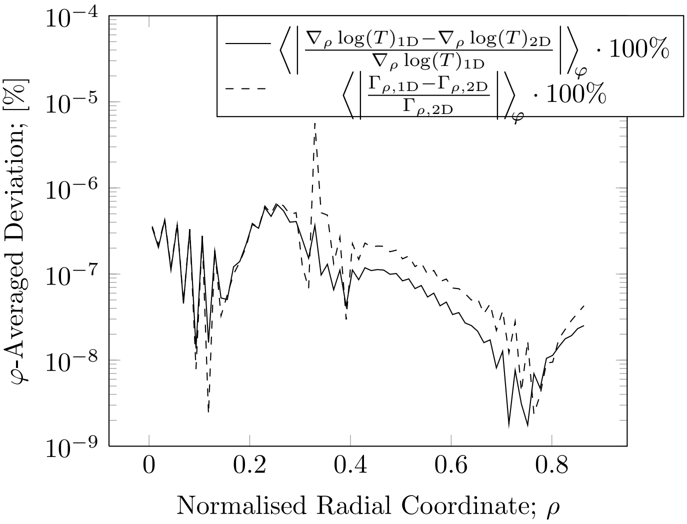


Figure 3.23: Deviation between steady-state temperature profiles for the 1D- and 2D implementations, averaged over the φ dimension; $\left\langle \left| \frac{T_{1\mathrm{D}} - T_{2\mathrm{D}}}{T_{1\mathrm{D}}} \right| \right\rangle_{\varphi} \cdot 100\%$. Radial resolution scan over 50, 60, 70, 80, 90 and 100 points, # φ fixed at 10, shown in (a). Poloidal scan for # $\rho : 70$ is shown in (b).

(a) Radial logarithmic gradient and total flux



(b) Poloidal logarithmic gradient and total flux

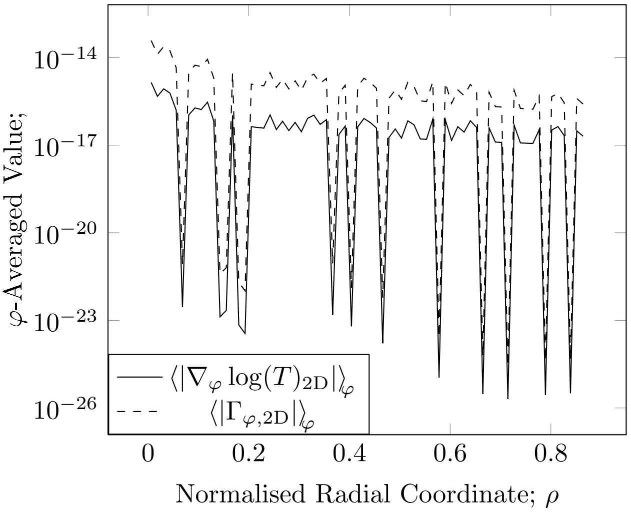
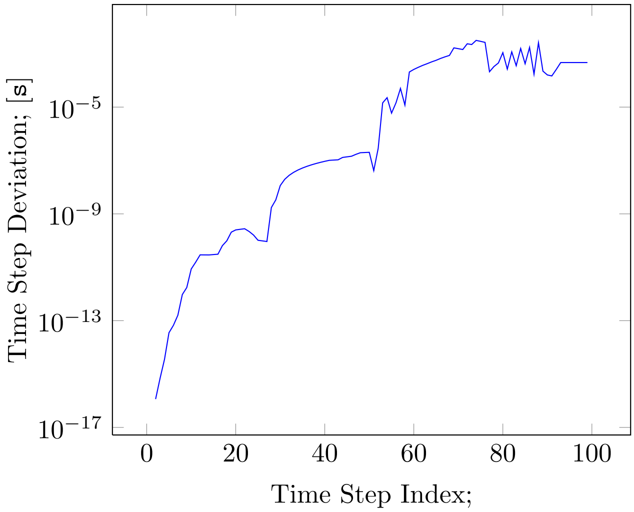


Figure 3.24: (a): Radial profile deviation between 1D and 2D implementation for the steady-state φ-averaged radial logarithmic temperature gradient and total heat flux (a). (b): Poloidal logarithmic temperature gradient and total heat flux radial profiles. *Spatial resolution*: $\rho \times \varphi = 70 \times 30$ (i.e. # ρ: 70 for 1D). *Temporal resolution*: RelTol=AbsTol=10^{-4}.

(a) Absolute value of time step deviation



(b) Time resolution scan; steady-state T-profile

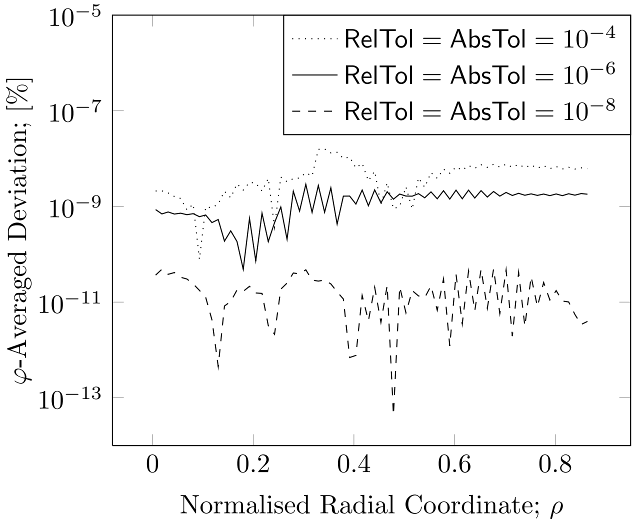
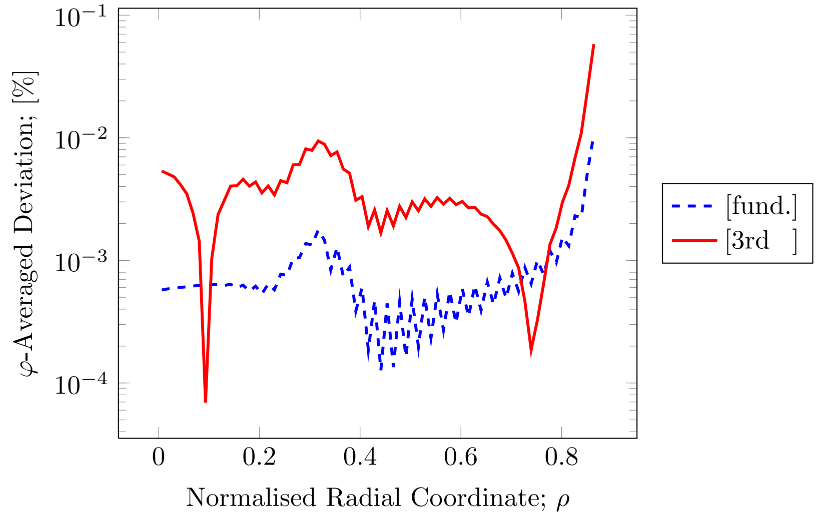


Figure 3.25: (a) is the absolute value of the deviation in numerical time steps taken for the 1D (# ρ: 70) and 2D (# $\rho \times \varphi = 70 \times 30$) implementations, at temporal resolution RelTol = AbsTol = 10^{-4}. (b) shows the deviation between steady-state temperature profiles for the 1D- and 2D implementations, averaged over the φ dimension; $\left\langle \left| \frac{T_{1\mathrm{D}} - T_{2\mathrm{D}}}{T_{1\mathrm{D}}} \right| \right\rangle_{\varphi} \cdot 100\%$, scanned over 3 temporal resolutions.

(a) Amplitude profiles: Deviation between 1D and 2D implementations



(b) Phase profiles: Deviation between 1D and 2D implementations

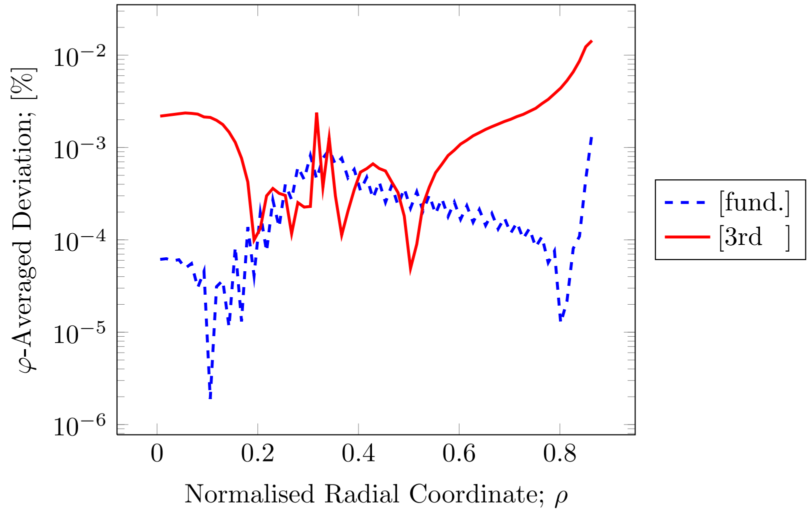


Figure 3.26: Deviations in heat modulation fundamental mode and 3rd harmonic amplitude (a) and phase profiles (b) for the 1D (# ρ: 70) and 2D (# $\rho \times \varphi = 70 \times 30$) implementations, respectively. The shown data is the φ-averaged deviation. *Source parameters*: $\mathcal{A}(t) = 0.75 \pm 0.55$. *Temporal resolution*: $\mathsf{RelTol} = \mathsf{AbsTol} = 10^{-4}$.

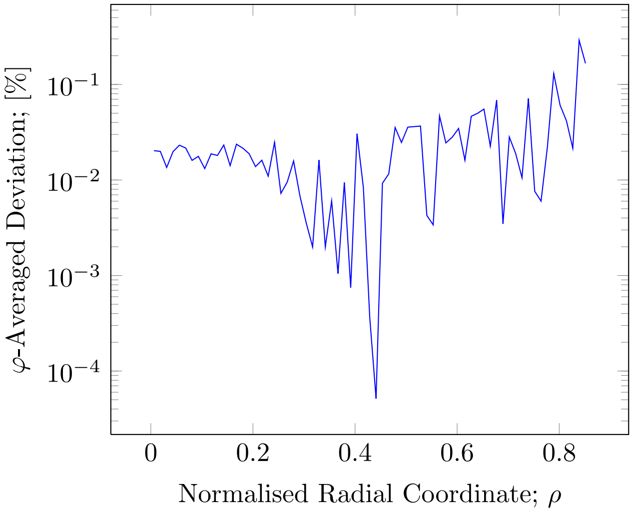


Figure 3.27: Deviation between cold pulse 30`eV` drop delay time for the 1D (# ρ: 70) and 2D (# $\rho \times \varphi = 70 \times 30$) implementations. *Temporal resolution*: RelTol = AbsTol = 10^{-4}.

The results indicate, that the origin of the deviation between 1D and 2D implementations is located somewhere within the ode23tb time-integrator source code. In lines 363 – 400 in the ode23tb source code the error is estimated using norms, and a rule of thumb is used to select the time step size. The norm is used to assign strictly positive lengths of the whole input array. Since the 1D and 2D input arrays differs in size and dimension, this might be the source of the deviation, as the 1D ode23tb input array has # φ times as few elements as the 2D ode23tb input array. If indeed the deviation between 1D and 2D results originates from the time step size estimation in ode23tb, the time steps taken through the time-integration should differ. Now, the number of time steps taken is the same for the 1D and 2D implementations when running simulations till 1`s`, at temporal resolution RelTol = AbsTol = 10^{-4}. However, Fig. 3.25 (a) shows that the size of each time step differs slightly in magnitude at the beginning, though more and more as time progresses. Note that the time step deviation is 0 at the end time which cannot be plotted in a logarithmic plot. Note also, the peculiar flatness of the deviation profile at $4.7 \cdot 10^{-4}$`s` for the last 7 indicies. The finite deviation displayed in Fig. 3.25 (a) supports the view that the deviation between results from 1D and 2D implementations arises due to the internal workings of the ode23tb solver, and not due to errors in the developed 2D script. If the temporal resolution is increased to RelTol = AbsTol = 10^{-6} and RelTol = AbsTol = 10^{-8}, the number of time steps taken for the 1D- and

2D steady-state simulations differ; 277 (1D) and 287 (2D) time steps for the former, 1115 (1D) and 1116 (2D) time steps for the latter temporal resolution. Fig. 3.25 (b) demonstrates that the deviation between steady-state radial temperature profile for the 1D and 2D implementations decreases as a function of temporal resolution. This further substantiates the established notion that deviation between 1D and 2D results originates from the `ode23tb` solver, and demonstrates that these deviations can be decreased by increasing the temporal resolution.

Figs. 3.26 and 3.27 show the deviation between heat modulation and cold pulse results generated by the 1D and 2D implementations for 70 radial grid points, and 30 poloidal grid points in the 2D implementation. Based on the previous results, these deviations are assessed to be within the acceptable range.

The results presented in this section substantiates the developed 2D implementation and it is therefore considered verified, thus enabling further investigations.

3.7.3 Outboard ICRH: Asymmetric Source Modeling

Having established a poloidally symmetric 2D model producing credible results, the source can now be altered to accommodate for the outboard localised ICRH heating employed in JET shot 55809. A simple Gaussian function,

$$G(\varphi) = \exp\left(-4[\varphi - \pi]^2\right), \tag{3.18}$$

is used to mimic the spatial localisation of the ICRH source in the poloidal dimension. Fig. 3.28 illustrates the degree of ICRH poloidal dimension localisation chosen. For the $G(\varphi)$ parameters chosen, Fig. 3.28 shows that ICRH heating is limited to the poloidal region $\varphi = \pi \pm 1$ for asymmetric source modeling experiments.

The total heating induced by the ICRH source is required to remain the same as that used in section 3.7.2 for the ith ρ grid point;

$$\int_0^{2\pi} \partial\varphi \;\; \mathcal{ICRH}_{\text{asym}}(\rho_i, \varphi) = \int_0^{2\pi} \partial\varphi \;\; \mathcal{ICRH}_{\text{sym}}(\rho_i) = 2\pi \mathcal{ICRH}_{\text{sym}}(\rho_i). \tag{3.19}$$

Then, the results of the implementation with asymmetric source can be compared directly to the results of the implementation with the symmetric source, which itself in section 3.7.2 is verified against the 1D implementation. Using the requirement stated in Eq. (3.19) and the poloidally localised source assumption of Eq. (3.18), an expression can be derived for $\mathcal{ICRH}_{\text{asym}}(\rho_i, \varphi_j)$ at the ith ρ- and jth φ grid points.

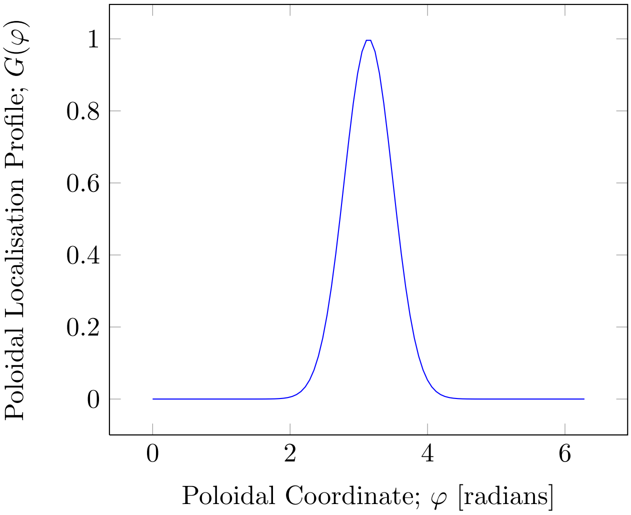


Figure 3.28: The Gaussian function, Eq. (3.18), mimicking ICRH poloidal localisation.

Expressing the ratio of the RHS of Eq. (3.19) and the poloidal integral over the RHS of Eq. (3.18),

$$\Xi(\rho_i) = \frac{2\pi\mathcal{ICRH}_{\mathrm{sym}}(\rho_i)}{\int_0^{2\pi}\partial\varphi \ \ \exp(-4[\varphi-\pi]^2)}, \tag{3.20}$$

the asymmetric ICRH heating defined by Eqs. (3.18-3.20) is

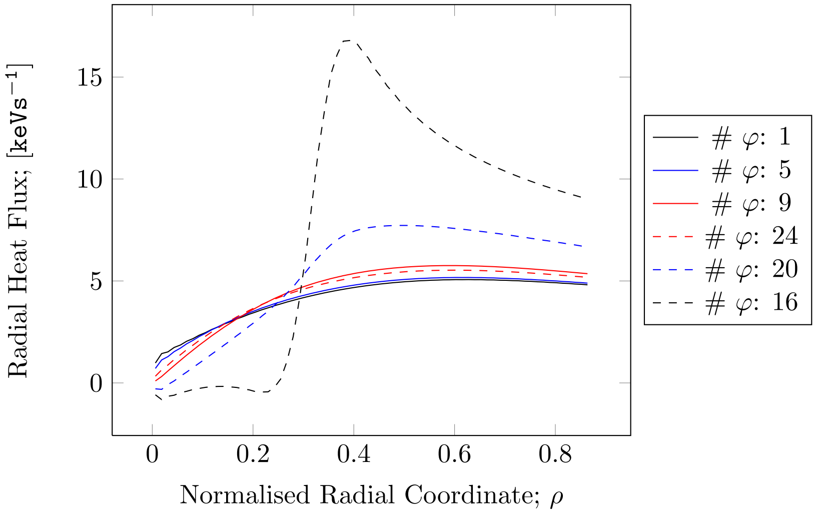
$$\mathcal{ICRH}_{\mathrm{asym}}(\rho_i,\varphi_j) = \Xi(\rho_i)G(\varphi_j) \tag{3.21}$$

at the ith radial- and jth poloidal grid point.

The physical picture of the restriction, Eq. (3.19), is that the total inflow of heating power deposited in the plasma per unit time is of equal magnitude for the implementations of symmetric and asymmetric ICRH. If the same amount of heat flux enters the plasma via the source term for the two implementations, both implementations must yield the same outwards directed radial flux in steady-state, when adding contributions for all φ. In this way, the asymmetric ICRH implementation can be verified through the requirement of radial heat flux conservation going from the symmetric to the asymmetric ICRH heating implementation.

Notice in Fig. 3.29 (a), that the φ-averaged total radial heat flux tends to go symmetric for $\rho \to 0$. This follows from the radial boundary condition at

(a) Symmetric to asymmetric source heat flux conservation



(b) Steady-state outboard & inboard plasma temperature profiles

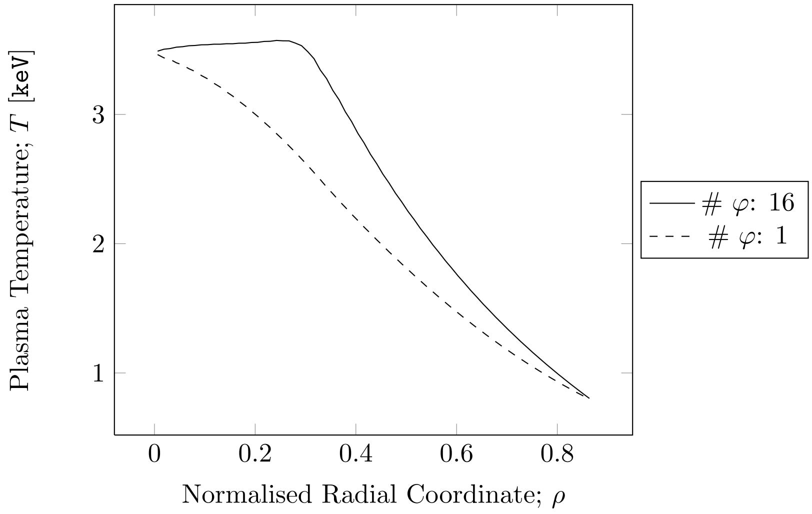
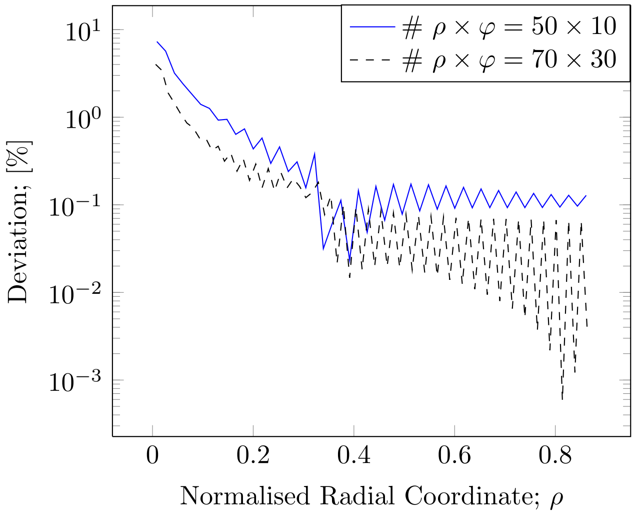


Figure 3.29: (a): Asymmetric heating steady-state radial heat fluxes for symmetric φ values. (b): Outboard ($\varphi = \pi$, # φ: 16) and inboard ($\varphi = 0$, # φ: 1) plasma temperature profiles. *Spatial resolution*: # $\rho \times \varphi = 70 \times 30$. *Temporal resolution*: $\mathsf{RelTol} = \mathsf{AbsTol} = 10^{-8}$.

(a) Radial heat flux conservation



(b) Total heat flux in the poloidal dimension; $\# \ \rho \times \varphi = 70 \times 30$

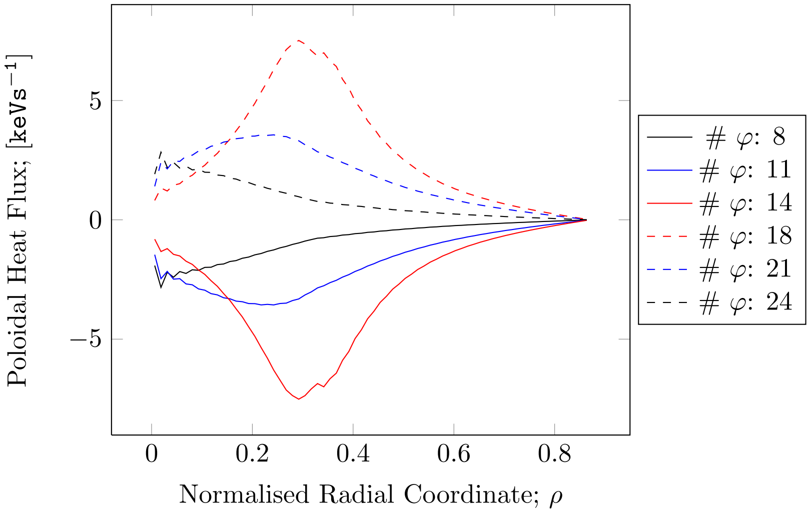


Figure 3.30: (a): Deviation between the poloidal dimensional sum over total heat flux radial profiles, $\left\langle \sum_{j=1}^{N_\varphi} \Gamma(\rho, \varphi_j) \right\rangle_{\mathcal{S}_{\text{asym}}}^{\mathcal{S}_{\text{sym}}}$, for the symmetric and asymmetric source implementations, respectively. (b): Poloidal heat fluxes for asymmetric heating. *Temporal resolution*: $\mathsf{RelTol} = \text{AbsTol} = 10^{-8}$.

$\rho = 0$; the radial plasma centre ghost points are set equal to the physical $\rho = 0$ value a poloidal distance $\Delta\varphi = \pi$ away. Thus, radial heat flux contributions should cancel out symmetrically in the total radial heat flux poloidal sum for each physical $\rho = 0$ pair a poloidal distance $\Delta\varphi = \pi$ apart. The outboard and inboard plasma temperature profiles shown in Fig. 3.29 (b) illustrate well the flux profiles in Fig. 3.29 (a). Notice, that the radial location of the outboard $\varphi = \pi$ ($\#\ \varphi$: 16) flux peak in Fig. 3.29 (a) corresponds to the radial location in Fig. 3.29 (b) featuring the steepest temperature gradient on the $\rho \in [0, 0.87]$ interval.

Fig. 3.30 (a) shows the deviation in the radial profile for the total radial heat flux between symmetric and asymmetric ICRH implementations accummulated over the poloidal dimension. Notice, that even though deviations do drop going from plasma centre to edge, the deviation exceeds 1% in the centre for both spatial resolutions. This is due to the fact, that all heat fluxes approach 0 in the plasma centre for the symmetric source implementation (homogeneous Neumann at $\rho = 0 \rightarrow 0$ temperature gradient $\rightarrow 0$ heat flux), thus creating great uncertainty in the deviation percentage which is normalised with symmetric heating fluxes in the denominator. The temporal resolution does not affect the deviation much ($\mathsf{RelTol} = \mathsf{AbsTol} = 10^{-4}$, 10^{-6} and 10^{-8} have been tried), however, as Fig. 3.30 (a) demonstrates, the deviation drops as a function of spatial resolution. Furthermore, it must be kept in mind that enforcing the requirement, Eq. (3.19), in practice, involves numerical integration of $G(\varphi)$, thus introducing numerical errors. Based on the results and interpretations in this section, it is assessed that the radial deviation profile is low enough to demonstrate radial flux conservation going from the symmetric to the asymmetric implementation. Thus, the asymmetric source implementation is considered verified.

In 1D CGM modeling experiments the 1D source profile, Eqs. (3.1-3.3), is assumed to constitute the average heating over all poloidal grid points. Therefore the requirement, Eq. (3.19), must be met for CGM numerical experiments conducted with asymmetric heating in order to be comparable to 1D results. However, the results for the asymmetric heating presented in this section are not quite physical, as a strong additional diffusion is present in the poloidal dimension in an actual 3D fusion plasma [29]. This extra diffusion acts to poloidally 'smear out' the deviation between outboard and inboard profiles observed for the steady-state temperature profiles shown in Fig. 3.29 (b).

3.7.4 Additional Poloidal Diffusion

The parallel dispersion poloidal component of the diffusion that occurs along the magnetic field lines has thus far been ignored. This diffusion coefficient is proportional to the poloidal velocity of the plasma species [1], and can be expressed as [29]

$$\chi_{\mathrm{pol}} = C_{\mathrm{pol}} \frac{1}{\rho} \left| \frac{\partial}{\partial \varphi} \log(T) \right| , \tag{3.22}$$

where C_{pol} adjusts the magnitude of poloidal diffusion. Values of C_{pol} approximating fusion plasma conditions are rather high ($C_{\mathrm{pol}} > 1000$) [1].

With this, the 2D CGM becomes

$$\begin{aligned} \frac{\partial T}{\partial t} = \nabla \cdot q^{\frac{3}{2}} T^{\frac{5}{2}} & \begin{bmatrix} \chi_s \Big[-R\frac{\partial}{\partial \rho} \log(T) - \kappa_c \Big] \mathsf{H}\Big(-R\frac{\partial}{\partial \rho} \log(T) - \kappa_c \Big) + \chi_0 \\ \chi_s \Big[-R\frac{\partial}{\partial \rho} \log(T) - \kappa_c \Big] \mathsf{H}\Big(-R\frac{\partial}{\partial \rho} \log(T) - \kappa_c \Big) + \chi_0 + \chi_{\mathrm{pol}} \end{bmatrix} \\ & \times \begin{bmatrix} \frac{\partial}{\partial \rho} \log(T) \\ \frac{1}{\rho}\frac{\partial}{\partial \varphi} \log(T) \end{bmatrix} + \mathcal{S} \end{aligned} \tag{3.23}$$

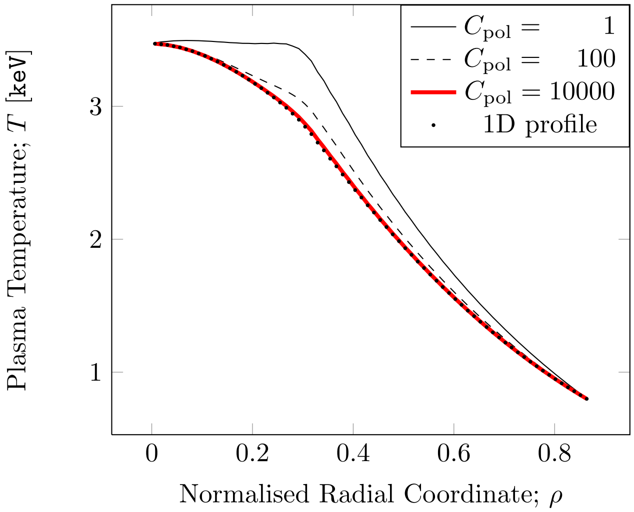
The additional poloidal diffusion acts to 'drain' the outboard temperature profile ICRH peak observed in Fig. 3.29 (b), evening out the temperature profile poloidally. Thus, for large enough C_{pol}, the temperature profile approaches that obtained with the 1D implementation, Fig. 3.16 (a). To achieve this effect, a slight modification has to be made to the poloidal localisation function, Eq. (3.18), such that $G(\varphi)$ is instead

$$G(\varphi) = \exp\left(-4 \left[\varphi + \frac{\delta\varphi}{2} - \pi \right]^2 \right) . \tag{3.24}$$

In this form, the Gaussian peak has been shifted half an inter-φ grid length, such that no poloidal derivatives are evaluated at the peak. Before this change, there was large poloidal diffusion in the vicinity of the $\varphi = \pi$ (outboard) radial profile, though practically no poloidal diffusion at φ itself, due to the symmetry of the Gaussian.

The steady-state temperature profiles for $C_{\mathrm{pol}} = 1$, 100 and 10000 are shown in Fig. 3.31 (a) for $\#\ \rho \times \varphi = 70 \times 30$, along with the steady-state temperature profile generated by the 1D implementation ($\#\ \rho$: 70). Observe, that the outboard temperature profile ICRH peak in Fig. 3.31 (a) diminishes as C_{pol} increases. This supports the expected behaviour that the 2D temperature profile approaches the 1D temperature profile as $C_{\mathrm{pol}} \to \infty$ (i.e. the localised source, $\mathcal{ICRH}_{\mathrm{asym}}$, is infinitely smeared out poloidally, thus approaching $\mathcal{ICRH}_{\mathrm{sym}}$ at all φ_j). The deviation between the 1D and 2D steady-state temperature profiles, for $C_{\mathrm{pol}} = 10000$, is shown in Fig. 3.31 (b). Notice that the peak deviation is in the vicinity of $\rho = \rho_{\mathrm{ICRH}}$ (0.32), as expected.

(a) Outboard temperature profile for various C_{pol}



(b) Deviation between 1D and 2D profiles for C_{pol}=10000

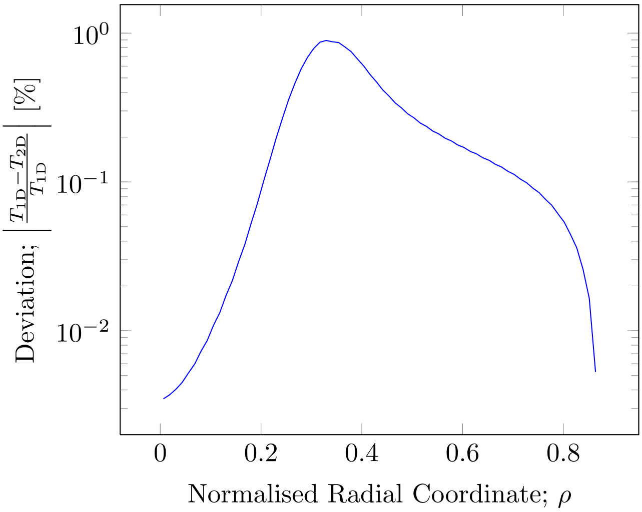
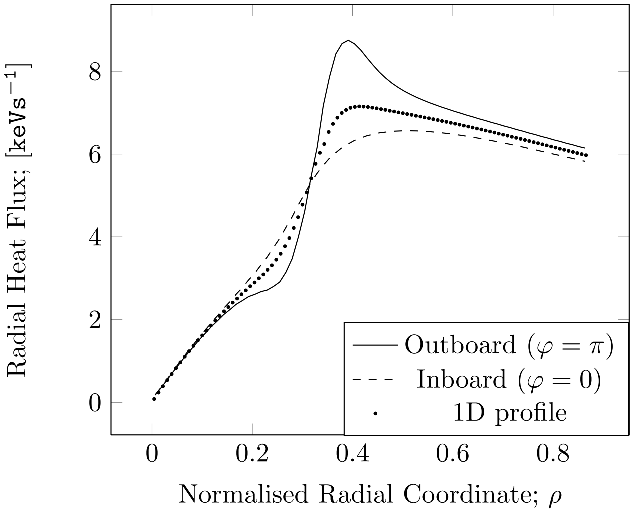


Figure 3.31: (a): Steady-state plasma temperature profiles predicted by the 1D CGM, Eq. (1.25), compared to the outboard ($\varphi = \pi$, # φ: 16) 2D CGM, Eq. (3.23). (b) Deviation between the 1D and 2D outboard $C_{\mathrm{pol}} = 10000$ profiles. *Spatial resolution*: # ρ: 70 (1D) and # $\rho \times \varphi = 70 \times 30$. *Temporal resolution*: $\mathsf{RelTol} = \mathsf{AbsTol} = 10^{-4}$.


(a) Radial heat flux for $C_{\mathrm{pol}} = 1000$

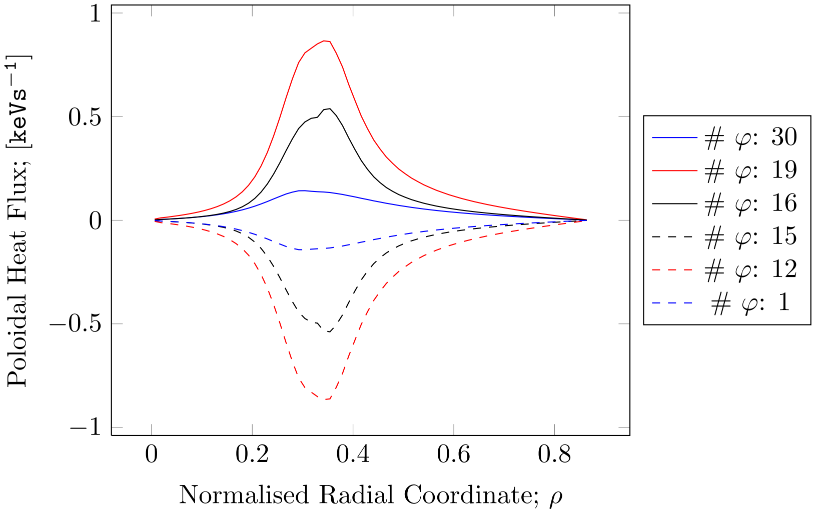

(b) Poloidal heat flux for C_{pol}=1000

Figure 3.32: Steady-state radial- (a) and poloidal (b) heat flux profiles for $C_{\mathrm{pol}} = 1000$. The radial heat fluxes are compared to the 1D case shown in Fig. 3.18 (b). *Spatial resolution*: # ρ: 70 (1D) and # $\rho \times \varphi = 70 \times 30$. *Temporal resolution*: $\mathsf{RelTol} = \mathsf{AbsTol} = 10^{-4}$.

The steady-state radial and poloidal heat fluxes are shown in Fig. 3.32. As observed in Fig. 3.32 (a), the radial heat flux agrees well with that of the 1D case, i.e. the heat flux from the 1D implementation is in the middle of the outboard and inboard flux profiles. The poloidal fluxes shown in Fig. 3.32 (b) display the symmetry expected when using Eq. (3.24) to localise the ICRH heating. Furthermore, the poloidal logarithmic temperature gradients are observed to be greater around the outboard profile in simulations for lower C_{pol}, in agreement with expectation; less poloidal diffusion means greater magnitude of poloidal logarithmic temperature gradient.

The results for the modulation of the asymmetric ICRH heating source are shown in Fig. 3.33. Notice, that the 2D amplitude and phase profiles at $\varphi = \frac{\pi}{2}$ are very close to the corresponding 1D results. As indicated in Figs. 3.31 and 3.32 (a), the asymmetric ICRH heating in Eq. (3.23) does not result in perfect outboard/inboard radial profile symmetry for $C_{\text{pol}} = 1000$. Therefore, it is not surprising to observe in Fig. 3.33 that the 2D $\varphi = \frac{\pi}{2}$ modulation results are closer to the 1D results, as the $\varphi = \frac{\pi}{2}$ radial temperature profile is expected to be close to the average between the outboard and inboard temperature profiles. No fundamentally different dynamics is detected in the heat modulation results, going from 1D radial to 2D poloidal plane simulations.

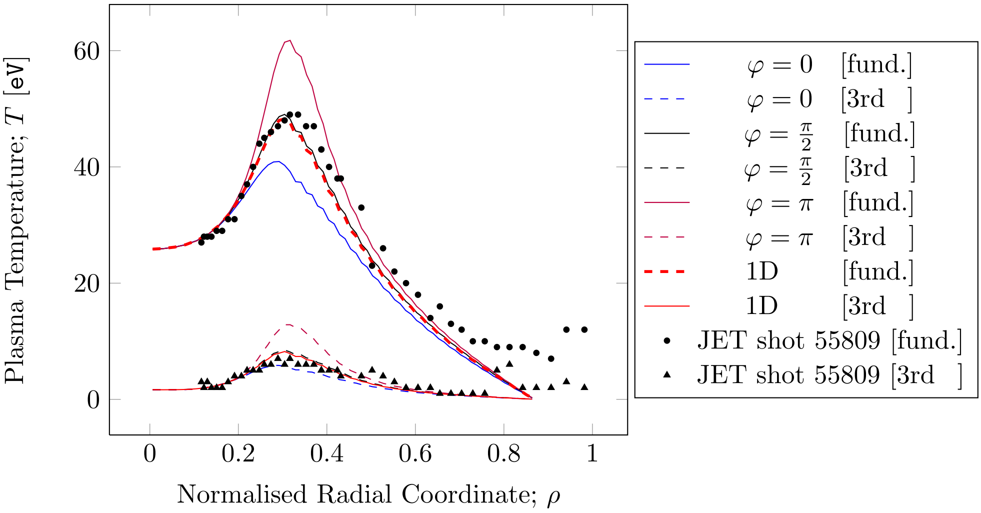
For 1D CGM modeling, the measured temperature at the plasma edge is assumed to be the average value for the whole plasma surface. Therefore, a poloidally constant average edge temperature must be kept during 2D cold pulse numerical experiments. This can be done by settling on a number of poloidal edge temperature grid points, say $N_{\text{cp}} = 3$, and apply the cold pulse edge data for them, while up-adjusting all other poloidal edge temperature grid points, thus keeping the φ-averaged edge temperature constant. Mathematically this can be expressed as

$$\begin{aligned} N_{\text{cp}}[T_{\text{ss}} - T_{\text{cp}}] &= [N_{\text{cp}} - N_{\varphi}][T_{\text{ss}} - T_{\neq\text{cp}}] \\ &\Downarrow \\ T_{\neq\text{cp}} &= \frac{N_{\text{cp}}[T_{\text{cp}} - T_{\text{ss}}]}{N_{\text{cp}} - N_{\varphi}} + T_{\text{ss}}, \end{aligned} \tag{3.25}$$

where T_{ss}, T_{cp} and $T_{\neq\text{cp}}$ are the steady-state, cold pulse temperature, and up-adjusted non-cold pulse plasma edge temperatures, respectively. N_{cp} and N_{φ} are the number of cold pulse grid points and the total poloidal resolution, respectively.

High C_{pol} values turns out to complicate cold pulse simulations as the `ode23tb` solver fails, supposedly due to having high poloidal flux at the sharp cold pulse/non-cold pulse inter-φ transition at the edge. However, with $C_{\text{pol}} = 1000$ cold pulse simulations can be run. In Fig. 3.34 cold pulse time traces are shown

(a) Amplitude profile: JET shot 55809 data compared to 1D & 2D CGM



(b) Phase profile: JET shot 55809 data compared to CGM

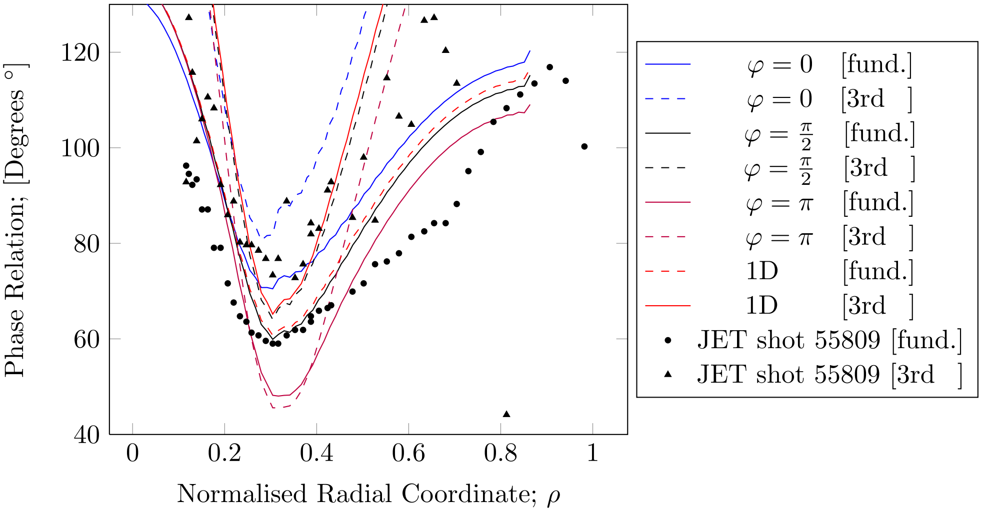
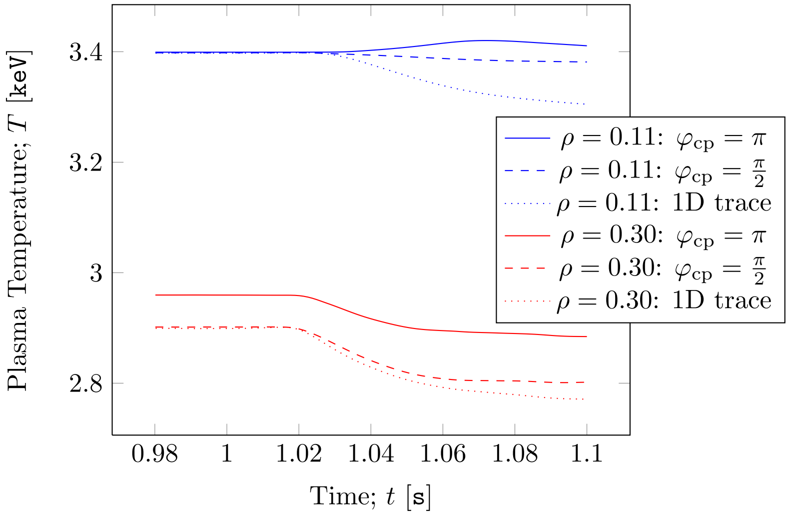


Figure 3.33: Heat modulation fundamental mode and 3rd harmonic amplitude and phase profiles predicted by the 1D CGM, Eq. (1.25), and 2D CGM, Eq. (3.23), compared to JET shot 55809 data. *Source parameters*: $\mathcal{A}(t) = 0.75 \pm 0.55$. *Spatial resolution*: $\#\ \rho$: 70 (1D) and $\#\ \rho \times \varphi = 70 \times 30$. *Temporal resolution*: $\mathsf{RelTol} = \mathsf{AbsTol} = 10^{-4}$.

(a) Cold pulse time traces for $\rho = 0.11$ and $\rho = 0.30$



(b) Cold pulse time traces for $\rho = 0.60$ and $\rho = 0.87$

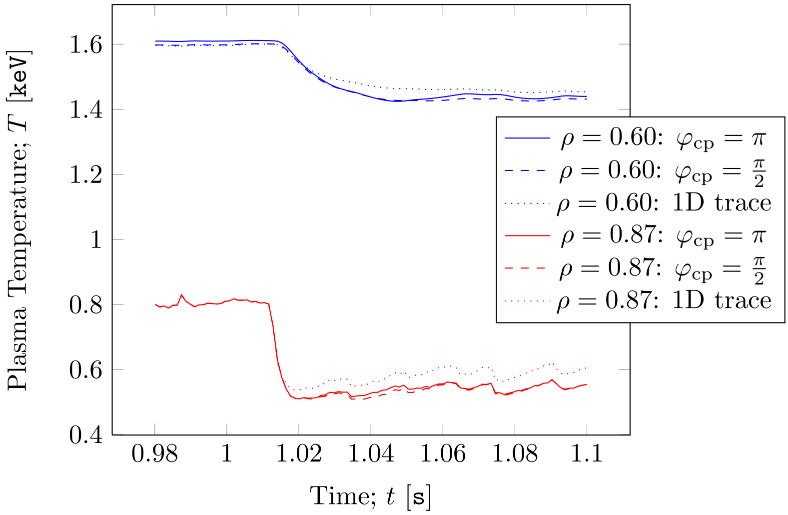


Figure 3.34: Cold pulse time traces modeled for $C_{\text{pol}} = 1000$ with the 2D CGM, Eq. (3.23), with cold pulses localised at $\varphi = \pi$ (outboard) and $\varphi = \frac{\pi}{2}$, respectively. *Spatial resolution*: # ρ: 100. *Temporal resolution*: $\mathsf{RelTol} = \mathsf{AbsTol} = 10^{-4}$.

for the cold pulse localised outboard ($\varphi = \pi$) and at $\varphi = \frac{\pi}{2}$, respectively. Notice the rather large deviation between the outboard profile and the 1D profile at $\rho = 0.30$, while the $\varphi = \frac{\pi}{2}$ profile and 1D profile are close. The reason for this local deviation is clear from Fig. 3.31, which demonstrates that the greatest difference between outboard and inboard temperature profiles is in the vicinity of $\rho_{\mathrm{ICRH}} = 0.32$. Notice also, that an increase in temperature in the core, $\rho = 0.11$, for the outboard time trace is present, whereas a slight drop is observed for the $\varphi = \frac{\pi}{2}$ time trace. This could be explained as a consequence of having poloidal fluxes. Fig. 3.32 (b) shows, that the outboard radial profile is a poloidal flux source profile, i.e. poloidal flux only runs to either poloidal side of the outboard profile (actually the true outboard profile is between # φ: 15 and 16). The localisation Gaussian depicted in Fig. 3.28 has steepest slopes in the vicinity of $\varphi = \pi$, which is shifted to $\varphi = \pi + \frac{\delta\varphi}{2}$ in Eq. (3.24). Since the asymmetric ICRH heating is proportional to this function in the poloidal dimension, the poloidal fluxes which are proportional to $\frac{1}{\rho}\frac{\partial}{\partial\varphi}\log(T)$ are greatest in the region around the outboard radial profile ($\varphi = \pi$), in particular in the plasma centre due to the inverse ρ proportionality. When the outboard localised cold pulse propagates towards the plasma centre it will thus be more exposed to thermal flux in the poloidal dimension, than the $\varphi = \frac{\pi}{2}$ localised cold pulse. All non cold pulse grid points at the plasma edge are up-adjusted according to Eq. (3.25), and thus induce warm pulses, each of amplitude a factor $\frac{N_{\mathrm{cp}}}{N_{\neq\mathrm{cp}}}$ smaller than the cold pulse. Due to the higher poloidal heat flux around the outboard profile, a greater fraction of warm pulse flows poloidally towards the outboard cold pulse and damps it, than for the $\varphi = \frac{\pi}{2}$ cold pulse. This could explain why the outboard cold pulse appears to be more damped than the $\varphi = \frac{\pi}{2}$ cold pulse in Fig. 3.34, and even goes above the pre-pulse temperature in the core.

CHAPTER 4

Arbitrary Geometry Modeling

Cylindrical symmetry has been assumed so far, for the poloidal profile of the plasma confined in the JET tokamak. The actual poloidal plasma profile observed in experiments is more D-shaped than round.

Within the present chapter, a numerical scheme is therefore presented, with which arbitrary geometries can be modeled. The chapter is organised as follows:

- Motivation for an alternative to the FDM.
- Restatement of the TSTM, stressing the model's conservation form.
- Presentation of a scheme allowing modeling of arbitrary geometries.
- Implementation of the 1D TSTM for this method
- Change of model coordinate system
- Implementation of the 2D TSTM.

4.1 Higher Dimensional Model Criteria

As argued for in section 2.1, working with finite differences for spatial discretisation has certain advantages. After all, the developed implementations outlined in chapter 2,

- have a fairly intuitive structure,
- appear to be producing trustworthy results,
- ensure flexibility in choice of time-integration method (due to the scheme's semidiscrete form).

However, for modeling in more than 1 spatial dimension several complications arise. The shortcomings listed below are caused by the implemented FDM's simple underlying structure;

- can only have simple dimension-by-dimension structure,
- non-grid-aligned boundaries are tricky to handle.

In general, FDMs are ill-suited to deal with complex geometries. If grid points are not equidistant in each dimension, finite difference based schemes are generally hard to implement.

Expanding the implementation basis presented in chapter 2 to 2D, as done in section 3.7, implies choosing one of the following options:

1. Stick with the cylindrical grid (as done in section 3.7).
2. Switch to cartesian grid.

Either choice has drawbacks. Choosing option 1. results in high resolution in the centre of the plasma torus, and low resolution at the plasma edge. This is undesirable since the cold pulse shock occurs at the plasma edge. Furthermore, future more detailed numerical studies, based on the approach used in section 3.7, might indicate that a symmetric cylindrical geometry is too much of a simplification when comparing to experiment.

With option 2. handling boundaries is nontrivial when the boundary conditions are non-grid-aligned, as is the case with Eqs. (4.8-4.10), unless one wants to model a (rather unphysical) square cross-section of a fusion plasma in 2D.

A model that goes beyond these limitations is needed.

4.2 Expressing the TSTM in Conservation Form

The TSTM [28] is formulated in cylindrical coordinates as:

$$\frac{\partial E}{\partial t} = \nabla \cdot \mathbf{\Gamma}_E + E\gamma\left(T, \nabla T\right) - [\gamma_0 + \beta E]E \tag{4.1}$$

$$\frac{\partial T}{\partial t} = \nabla \cdot \mathbf{\Gamma}_T + S(\rho, \phi, z) \tag{4.2}$$

where the turbulent energy $E = E(\rho, \phi, z)$ and electron temperature $T = T(\rho, \phi, z)$. The flux-functions $\mathbf{\Gamma}_E$ and $\mathbf{\Gamma}_T$ represent turbulent energy flux and thermal flux for the electron species, respectively. In spatial regions of the temperature profile where the critical gradient, κ_c, is exceeded, the otherwise small damping term $\gamma\left(T, \nabla T\right)E$ becomes a large positive growth term, representing a significant contribution to the build-up of turbulence in super-critical regions. The underlying instability, $\gamma\left(T, \nabla T\right)$, is expressed mathematically by Eqs. (1.26-1.27).

In the 1D radial version of the TSTM, presented in [28], flux-functions $\mathbf{\Gamma}_E$ and $\mathbf{\Gamma}_T$ are:

$$\mathbf{\Gamma}_E^{\mathrm{1D}} = D_0 E \nabla E \hat{\boldsymbol{\rho}} \tag{4.3}$$

$$\mathbf{\Gamma}_T^{\mathrm{1D}} = \left[\frac{3}{2}\chi_0 T^{\frac{5}{2}} q^{\frac{3}{2}} \frac{\nabla T}{T} - q_{\mathsf{h}}\left(E, T, \nabla T\right)\right] \hat{\boldsymbol{\rho}}. \tag{4.4}$$

In the 2D model, the flux-functions are here formulated as:

$$\mathbf{\Gamma}_E^{\mathrm{2D}} = D_0 E \nabla E \hat{\boldsymbol{\rho}} + [D_0 + D_{\mathrm{pol}}(T)] E \nabla E \hat{\boldsymbol{\phi}} \tag{4.5}$$

$$\begin{aligned} \mathbf{\Gamma}_T^{\mathrm{2D}} = & \left[\frac{3}{2}\chi_0 T^{\frac{5}{2}} q^{\frac{3}{2}} \frac{\nabla T}{T} - q_{\mathsf{h}}\left(E, T, \nabla T\right)\right] \hat{\boldsymbol{\rho}} \\ & + \left[\frac{3}{2}\chi_0 T^{\frac{5}{2}} q^{\frac{3}{2}} \frac{\nabla T}{T} - q_{\mathsf{h}}\left(E, T, \nabla T\right) + D_{\mathrm{pol}}(T)\right] \hat{\boldsymbol{\phi}}, \end{aligned} \tag{4.6}$$

i.e. diffusion in the poloidal dimension is increased by the poloidal diffusion factor, $D_{\mathrm{pol}}(T)$.

The underlying instability, γ, and turbulent part of the thermal flux , q_{h}, are:

$$q_{\mathsf{h}}\left(E, T, \nabla T\right) = CET \tanh\big(\gamma\left(E, T, \nabla T\right)\big) \tag{4.7}$$

The boundary conditions are:

$$\left.\frac{\partial E}{\partial \rho}\right|_{\rho=0} = \left.\frac{\partial T}{\partial \rho}\right|_{\rho=0} = 0 \qquad \text{i.e.} \qquad \text{smooth profiles (1D only)}, \tag{4.8}$$

$$\nabla \cdot \mathbf{\Gamma}_E = 0 \qquad \text{at} \qquad \partial S_{\text{tube}}, \tag{4.9}$$

$$T(R, \phi, z) = 0.8\texttt{keV} \qquad \text{(i.e. at } \partial S_{\text{tube}}), \tag{4.10}$$

where ∂S_{tube} denotes the surface of the 3D plasma tube. Boundary condition (4.9) states, that turbulence spreading in the plasma is constant at the edge. Boundary condition (4.10) enforces an inhomogeneous Dirichlet condition on the plasma edge. The assigned edge temperature is based on experimental data from JET.

4.3 Nodal Discontinuous Galerkin Method

The dicontinuous Galerkin finite element method (DG-FEM) is a combination of the finite element- (FEM) and finite volume methods (FVM), utilising a space of basis and test functions that mimics the FEM, but satisfies the equation modeled in a sense closer to the FVM. Locality is ensured in the DG-FEM by duplicating variables located at element nodes (i.e. element boundary values are duplicated). The theory presented in this section is mainly based on the book by Jan Hesthaven and Tim Warburton [12].

As shown in section 4.2, the problem at hand can be expressed as a conservation law in differential form;

$$\frac{\partial \mathbf{U}}{\partial t} = \nabla \cdot \mathbf{\Gamma} + \mathbf{S}, \tag{4.11}$$

In each finite element, the solution of the conserved quantities,

$$\mathbf{U}(\mathbf{x}, t) = \begin{cases} U_1(\mathbf{x}, t) \\ U_2(\mathbf{x}, t) \end{cases} \tag{4.12}$$

is assumed to be of the form

$$\mathbf{U}(\mathbf{x}, t) \approx \mathbf{U}_h(\mathbf{x}, t) = \bigoplus_{k=1}^{K} \mathbf{U}_h^k(\mathbf{x}, t) \in \mathsf{V}_h = \bigoplus_{k=1}^{K} \left\{ \psi_n^k \left(\mathsf{D}^k \right) \right\}_{n=1}^{N_p},$$

where $\bigoplus_{k=1}^{K}$ is the direct sum operator over elements $k = 1, 2, \cdots, K$, V_h denotes the space of the N_p piecewise polynomial basis functions ψ_n, defined on the space of element k, D^k. The local solution for each element is then

$$\mathbf{x} \in \mathsf{D}^k \ : \ \mathbf{U}_h^k(\mathbf{x}, t) = \sum_{i=1}^{N_p} \mathbf{U}_h^k(\mathbf{x}_i, t)\ell_i^k(\mathbf{x}) \in \mathsf{V}_h, \tag{4.13}$$

$\ell_i^k(\mathbf{x})$ being the multidimensional interpolating Lagrange polynomial,

$$\ell_i(\mathbf{x}) = \prod_{\substack{j=1 \\ i \neq j}}^{N_p} \frac{\mathbf{x} - \mathbf{x}_j}{\mathbf{x}_i - \mathbf{x}_j} \tag{4.14}$$

defined by some grid points, $\mathbf{x}_i$, on element k. Note the property, $\ell_i(\mathbf{x}_j) = \delta_{ij}$. $\ell_i^k(\mathbf{x})$ are chosen to be the basis polynomials for both $U_1(\mathbf{x}, t)$ and $U_2(\mathbf{x}, t)$, defined in Eq. (4.12).

Similarly, for the flux-function and source

$$\mathbf{\Gamma}(\mathbf{x}, t, \mathbf{U}) = \left\{ \begin{array}{l} \mathbf{\Gamma}_1(\mathbf{x}, t, \mathbf{U}) \\ \mathbf{\Gamma}_2(\mathbf{x}, t, \mathbf{U}) \end{array} \right. , \qquad \mathbf{S}(\mathbf{x}, t, \mathbf{U}) = \left\{ \begin{array}{l} \mathbf{S}_1(\mathbf{x}, t, \mathbf{U}) \\ \mathbf{S}_2(\mathbf{x}, t, \mathbf{U}) \end{array} \right. ,$$

it is assumed, that

$$\begin{aligned} \mathbf{\Gamma}(\mathbf{x}, t, \mathbf{U}) \approx \mathbf{\Gamma}_h(\mathbf{x}, t, \mathbf{U}) &= \bigoplus_{k=1}^{K} \mathbf{\Gamma}_h^k(\mathbf{x}, t, \mathbf{U}) \in \mathsf{V}_h = \bigoplus_{k=1}^{K} \left\{ \psi_n^k \left(\mathsf{D}^k\right) \right\}_{n=1}^{N_p}, \\ \mathbf{S}(\mathbf{x}, t, \mathbf{U}) \approx \mathbf{S}_h(\mathbf{x}, t, \mathbf{U}) &= \bigoplus_{k=1}^{K} \mathbf{S}_h^k(\mathbf{x}, t, \mathbf{U}) \in \mathsf{V}_h = \bigoplus_{k=1}^{K} \left\{ \psi_n^k \left(\mathsf{D}^k\right) \right\}_{n=1}^{N_p}. \end{aligned}$$

Analogous to Eq. (4.13);

$$\mathbf{x} \in \mathsf{D}^k \ : \ \mathbf{\Gamma}_h^k(\mathbf{x}, t) = \sum_{i=1}^{N_p} \mathbf{\Gamma}_h^k(\mathbf{x}_i, t)\ell_i^k(\mathbf{x}) \in \mathsf{V}_h, \tag{4.15}$$

$$\mathbf{x} \in \mathsf{D}^k \ : \ \mathbf{S}_h^k(\mathbf{x}, t) = \sum_{i=1}^{N_p} \mathbf{S}_h^k(\mathbf{x}_i, t)\ell_i^k(\mathbf{x}) \in \mathsf{V}_h. \tag{4.16}$$

With assumptions (4.13), (4.15) and (4.16), the local residual

$$\mathcal{R}_h = \left\{ \begin{array}{l} \mathcal{R}_{h,1} \\ \mathcal{R}_{h,2} \end{array} \right.$$

can be written,

$$\mathcal{R}_h = \frac{\partial \mathbf{U}_h^k}{\partial t} - \nabla \cdot \mathbf{\Gamma}_h^k - \mathbf{S}_h^k. \tag{4.17}$$

The residual is required to be orthogonal to all basis functions, i.e. ℓ_i^k

$$\int_{\mathsf{D}^k} \mathrm{d}\mathbf{x}\ \ell_i^k \mathcal{R}_h = 0. \tag{4.18}$$

Inserting Eq. (4.17) in Eq. (4.18),

$$\int_{\mathsf{D}^k} \mathrm{d}\mathbf{x}\ \ell_i^k \frac{\partial \mathbf{U}_h^k}{\partial t} - \ell_i^k \nabla \cdot \mathbf{\Gamma}_h^k - \ell_i^k \mathbf{S}_h^k = 0, \tag{4.19}$$

and integrating the middle term by parts, results in

$$\int_{\mathsf{D}^k} \mathrm{d}\mathbf{x}\ \ell_i^k \nabla \cdot \mathbf{\Gamma}_h^k = \int_{\partial\mathsf{D}^k} \mathrm{d}\mathbf{x}\ \hat{\boldsymbol{n}} \cdot \mathbf{\Gamma}_h^k \ell_i^k - \int_{\mathsf{D}^k} \mathrm{d}\mathbf{x}\ \mathbf{\Gamma}_h^k \cdot \nabla \ell_i^k. \tag{4.20}$$

The surface integral on the RHS is calculated exclusively on element-element interfaces. However, the solution $\mathbf{U}_h^k$ is doubly defined there, so one must find some way to estimate the flux, $\mathbf{\Gamma}_h^k$. This estimate is called *the numerical flux*, denoted $\mathbf{\Gamma}^*$. Solving non-linear systems of hyperbolic conservation laws via the DG-FEM require this building block, and $\mathbf{\Gamma}^*$ should be chosen such that the scheme reduces to a monotone scheme in the low order/finite volume limit [12]. This is ensured by requiring that the numerical flux is nondecreasing in the first argument and nonincreasing in the second argument [18]. The role of the numerical flux is to guarantee stability of the formulation by mimicking the flow of information in the underlying partial differential equation and therefore the choice of numerical flux has profound influence on the properties of the resulting schemes.

The simplest choice of numerical flux is to combine information from both elements by taking flux-function averages over the two sides of the element-element boundaries, called *centred* or *symmetric scheme.* Utilising this symmetric stencil, one does not explicitly make use of wave propagation information contained in the differential equations in the construction of the numerical flux. This information can be obtained as the solution to the Riemann problem, and numerical fluxes make use of this are called *upwind* or *Godunov-type methods* [36].

In general, the Lax-Friedrichs flux is used [37],

$$\mathbf{\Gamma}^* = \{\{\mathbf{\Gamma}_h\}\} + \frac{1}{2} \max_{\mathbf{U}} \left| \lambda \left(\hat{\boldsymbol{n}} \cdot \frac{\partial \mathbf{\Gamma}}{\partial \mathbf{U}} \right) \right| [\![\mathbf{U}_h]\!], \tag{4.21}$$

where

$$\{\{\mathbf{\Gamma}_h\}\} = \frac{\mathbf{\Gamma}_h^- + \mathbf{\Gamma}_h^+}{2} \qquad \text{and} \qquad [\![\mathbf{U}_h]\!] = \hat{\boldsymbol{n}}^- \cdot \mathbf{U}_h^- + \hat{\boldsymbol{n}}^+ \cdot \mathbf{U}_h^+, \tag{4.22}$$

'$-/+$' superscripts denotes 'interior/exterior' information, and $\lambda\left(\hat{\boldsymbol{n}} \cdot \frac{\partial \mathbf{\Gamma}}{\partial \mathbf{U}}\right)$ are the eigenvalues of the matrix $\hat{\boldsymbol{n}} \cdot \frac{\partial \mathbf{\Gamma}}{\partial \mathbf{U}}$. The Lax-Friedrichs flux generally leads to efficient, accurate and robust methods [36].

Inserting (4.20) in Eq. (4.19), results in the *weak form* of the nodal discontinuous Galerkin scheme;

$$\int_{\mathsf{D}^k} \mathrm{d}\mathbf{x}\ \ell_i^k \frac{\partial \mathbf{U}_h^k}{\partial t} + \mathbf{\Gamma}_h^k \cdot \nabla \ell_i^k - l_i^k \mathbf{S}_h^k = \int_{\partial \mathsf{D}^k} \mathrm{d}\mathbf{x}\ \hat{\boldsymbol{n}} \cdot \mathbf{\Gamma}^* \ell_i^k. \tag{4.23}$$

Integration by parts once more, results in the *strong form* of the nodal discontinuous Galerkin scheme,

$$\int_{\mathsf{D}^k} \mathrm{d}\mathbf{x}\ \ell_i^k \frac{\partial \mathbf{U}_h^k}{\partial t} - \ell_i^k \nabla \cdot \mathbf{\Gamma}_h^k - \ell_i^k \mathbf{S}_h^k = \int_{\partial \mathsf{D}^k} \mathrm{d}\mathbf{x}\ \hat{\boldsymbol{n}} \cdot \left[\mathbf{\Gamma}^* - \mathbf{\Gamma}_h^k\right] \ell_i^k. \tag{4.24}$$

The weak and strong DG-FEM schemes are mathematically equivalent - but computationally different.

Defining the mass matrix $\mathcal{M}$ and stiffness matrix $\mathcal{S}$,

$$\mathcal{M}_{ij}^k \equiv \int_{\mathsf{D}^k} \mathrm{d}\mathbf{x}\ \ell_i^k(\mathbf{x}) \ell_j^k(\mathbf{x}), \quad \mathcal{S}_{ij}^k \equiv \int_{\mathsf{D}^k} \mathrm{d}\mathbf{x}\ \left[\nabla \ell_i^k(\mathbf{x})\right] \ell_j^k(\mathbf{x}), \tag{4.25}$$

Eq. (4.23) can be written in index notation as

$$\mathcal{M}_{ij}^k \frac{\partial \mathbf{U}_h^k(\mathbf{x}_j, t)}{\partial t} + \mathcal{S}_{ij}^k \mathbf{\Gamma}_h^k(\mathbf{x}_j, t) - \mathcal{M}_{ij}^k \mathbf{S}_h^k(\mathbf{x}_j, t) = \int_{\partial \mathsf{D}^k} \mathrm{d}\mathbf{x}\ \hat{\boldsymbol{n}} \cdot \mathbf{\Gamma}^* \ell_i^k(\mathbf{x}). \tag{4.26}$$

Defining

$$\vec{\mathbf{U}}_h^k(t) \equiv \left\{ \begin{array}{c} \begin{pmatrix} U_{h,1}^k(\mathbf{x}_1, t) \\ U_{h,1}^k(\mathbf{x}_2, t) \\ \vdots \\ U_{h,1}^k(\mathbf{x}_{N_p}, t) \end{pmatrix} \\ \\ \begin{pmatrix} U_{h,2}^k(\mathbf{x}_1, t) \\ U_{h,2}(\mathbf{x}_2, t) \\ \vdots \\ U_{h,2}(\mathbf{x}_{N_p}, t) \end{pmatrix} \end{array} \right., \qquad \vec{\mathbf{\Gamma}}_h^k(t) \equiv \left\{ \begin{array}{c} \begin{pmatrix} \mathbf{\Gamma}_{h,1}^k(\mathbf{x}_1, t) \\ \mathbf{\Gamma}_{h,1}^k(\mathbf{x}_2, t) \\ \vdots \\ \mathbf{\Gamma}_{h,1}^k(\mathbf{x}_{N_p}, t) \end{pmatrix} \\ \\ \begin{pmatrix} \mathbf{\Gamma}_{h,2}^k(\mathbf{x}_1, t) \\ \mathbf{\Gamma}_{h,2}(\mathbf{x}_2, t) \\ \vdots \\ \mathbf{\Gamma}_{h,2}(\mathbf{x}_{N_p}, t) \end{pmatrix} \end{array} \right.,$$

$$\vec{\mathbf{S}}^k_h(t) \equiv \left\{ \begin{array}{c} \begin{pmatrix} S^k_{h,1}(\mathbf{x}_1, t) \\ S^k_{h,1}(\mathbf{x}_2, t) \\ \vdots \\ S^k_{h,1}(\mathbf{x}_{N_p}, t) \end{pmatrix} \\ \begin{pmatrix} S^k_{h,2}(\mathbf{x}_1, t) \\ S_{h,2}(\mathbf{x}_2, t) \\ \vdots \\ S_{h,2}(\mathbf{x}_{N_p}, t) \end{pmatrix} \end{array} \right. , \qquad \vec{\boldsymbol{\ell}^k} \equiv \begin{pmatrix} \ell^k_1(\mathbf{x}) \\ \ell^k_2(\mathbf{x}) \\ \vdots \\ \ell^k_{N_p}(\mathbf{x}) \end{pmatrix},$$

Eq. (4.26) is recast in matrix notation as

$$\mathcal{M}^k \frac{\partial \vec{\mathbf{U}}^k_h(t)}{\partial t} + \mathcal{S}^k \vec{\mathbf{\Gamma}}^k_h(t) - \mathcal{M}^k \vec{\mathbf{S}}^k_h(t) = \int_{\partial \mathsf{D}^k} \mathrm{d}\mathbf{x} \;\, \hat{\boldsymbol{n}} \cdot \mathbf{\Gamma}^* \vec{\boldsymbol{\ell}^k}(\mathbf{x})$$
$$\Updownarrow$$
$$\mathcal{M}^k \frac{\partial \vec{\mathbf{U}}^k_h(t)}{\partial t} = -\mathcal{S}^k \vec{\mathbf{\Gamma}}^k_h(t) + \mathcal{M}^k \vec{\mathbf{S}}^k_h(t) + \int_{\partial \mathsf{D}^k} \mathrm{d}\mathbf{x} \;\, \hat{\boldsymbol{n}} \cdot \mathbf{\Gamma}^* \vec{\boldsymbol{\ell}^k}(\mathbf{x}). \tag{4.27}$$

In this formulation, the first-order spatial derivative on the kth element can thus be expressed in terms of the differentiation matrix

$$\mathcal{D}^k = \left[\mathcal{M}^k\right]^{-1} \mathcal{S}^k. \tag{4.28}$$

The DG-FEM offers spatial grid flexibility, in that the resolution is controlled by both the number of elements and the order of the polynomial approximating the solution on each element. The local definition of the solution on one element makes the evaluation of solutions on each of the total N elements suitable for parallelisation on up to N cores, as is the case with simple finite difference. Comparing to the traditional FEM, the mass matrix is local in the DG-FEM, rather than global as in the FEM, and can thus be inverted at very little cost, yielding an explicit semidiscrete scheme. By carefully designing the numerical flux to reflect the underlying dynamics, one has more flexibility than in classical FEM to ensure stability for wave dominated problems, exactly the kind of tool needed for efficient CGM and TSTM modeling. Furthermore, the DG-FEM overcomes the key limitation of the FVM on achieving high-order accuracy on general grids by enabling this through the local element-based basis. This is all achieved while maintaining benefits such as local conservation and flexibility in the choice of numerical flux.

4.4 DG-FEM in 1 Dimension

The 1D TSTM is defined by inserting Eqs. (4.3-4.4) in Eqs. (4.1-4.2),

$$\frac{\partial E}{\partial t} = \nabla \cdot \Big[D_0 E \nabla E\Big]\hat{\boldsymbol{\rho}} + E\gamma\left(T, \frac{\partial T}{\partial \rho}\right) - [\gamma_0 + \beta E]E, \tag{4.29}$$

$$\frac{\partial T}{\partial t} = \nabla \cdot \left[\frac{3}{2}\chi_0 T^{\frac{5}{2}} q^{\frac{3}{2}} \frac{\nabla T}{T} - q_{\mathsf{h}}\left(E, T, \frac{\partial T}{\partial \rho}\right)\right]\hat{\boldsymbol{\rho}} + S(\rho), \tag{4.30}$$

where functions γ and q_{h} are given in Eqs. (1.26) and (4.7), respectively.

The model described by Eqs. (4.29-4.30) is subject to the boundary conditions (4.8-4.10).

At this point, a choice between the weak or the strong formulation of the DG-FEM must be made. The differences between the two schemes are the flux-function terms and RHSs of Eqs. (4.23) and (4.24). In the strong form, the divergence operator reemerges, whereas in the weak form, partial integration returns a gradient term. In cylindrical coordinates, the radial component of the divergence of some vector quantity involves a term with inverse radial dependence, i.e. a discontinuity is present. Thus, this issue is eliminated in the weak formulation, and the stiffness matrix remains simple to evaluate numerically. The trade-off is the loss of local conservation, as enforced in the strong formulation. This might cause significant numerical errors if the TSTM is dominated by convection terms, whereas the implementation should work if it is diffusion dominated. Based on the above discussion, the weak form is chosen.

To cast Eqs. (4.29-4.30) in the DG-FEM weak form, Eq. (4.27), the solutions, flux-functions and source terms are identified as

$$\mathbf{U}(\rho, t) = \begin{cases} E(\rho, t) \\ T(\rho, t) \end{cases}, \tag{4.31}$$

$$\boldsymbol{\Gamma}(\rho, t, \mathbf{U}) = \begin{cases} D_0 E \nabla E \hat{\boldsymbol{\rho}} \\ \left[\frac{3}{2}\chi_0 T^{\frac{5}{2}} q^{\frac{3}{2}} \frac{\nabla T}{T} - q_h\left(E, T, \nabla T\right)\right]\hat{\boldsymbol{\rho}} \end{cases}, \tag{4.32}$$

$$\mathbf{S}(\rho, t, \mathbf{U}) = \begin{cases} E\gamma\left(T, \nabla T\right) - [\gamma_0 + \beta E]E \\ S(\rho) \end{cases}, \tag{4.33}$$

respectively. Using Eq. (4.27), the weak form DG-FEM scheme for the TSTM in 1 dimension is thus;

$$\mathcal{M}^k \frac{\mathrm{d}\vec{\mathbf{U}}_h^k(t)}{\mathrm{d}t} = -\mathcal{S}^k \vec{\boldsymbol{\Gamma}}_h^k(t) + \mathcal{M}^k \vec{\mathbf{S}}_h^k(t) + \int_{\partial \mathsf{D}^k} \mathrm{d}\rho \;\, \hat{\boldsymbol{n}} \cdot \boldsymbol{\Gamma}^* \vec{\boldsymbol{\ell}}^k(\rho). \tag{4.34}$$

To approximate the gradients and explicit derivatives in Eqs. (4.32-4.33), the 1D TSTM is split in two first-order DG-FEM schemes. Denoting

$$\mathbf{Q} = \frac{\mathrm{d}\mathbf{U}}{\mathrm{d}\rho} = \left\{ \begin{array}{l} \frac{\mathrm{d}E(\rho,t)}{\mathrm{d}\rho} = Q_E(\rho,t) \\ \frac{\mathrm{d}T(\rho,t)}{\mathrm{d}\rho} = Q_T(\rho,t) \end{array} \right. , \tag{4.35}$$

and assuming

$$\mathbf{Q}(\rho,t) \approx \mathbf{Q}_h(\rho,t) = \bigoplus_{k=1}^{K} \mathbf{Q}_h^k(\rho,t) \in \mathsf{V}_h = \bigoplus_{k=1}^{K} \left\{ \psi_n^k \left(\mathsf{D}^k\right) \right\}_{n=1}^{N_p},$$

$$\rho \in \mathsf{D}^k \;:\; \mathbf{Q}_h^k(\rho,t) = \sum_{i=1}^{N_p} \mathbf{Q}_h^k(\rho_i,t)\ell_i^k(\rho) \in \mathsf{V}_h,$$

as for $\mathbf{U}$, the DG-FEM scheme corresponding to Eq. (4.35) is

$$\mathcal{M}^k \vec{\mathbf{Q}}_h^k(t) = \mathcal{S}^k \vec{\mathbf{U}}_h^k(t) - \int_{\partial \mathsf{D}^k} \mathrm{d}\rho \;\, \hat{\boldsymbol{n}} \cdot \mathbf{U}^* \vec{\boldsymbol{\ell}}^k(\rho), \tag{4.36}$$

analogous to that of Eq. (4.11), derived in detail in section 4.3. Eq. (4.36) is used to evaluate $\mathbf{U}_h^k$, $\mathbf{\Gamma}_h^k$ and $\mathbf{S}_h^k$ at the jth collocation point, ρ_j. That is; the numerical estimates $\mathbf{U}_h^k$, $\mathbf{\Gamma}_h^k$ and $\mathbf{S}_h^k$ of Eqs. (4.31-4.33) on element k, are evaluated numerically as

$$\mathbf{U}_h^k(\rho_j,t) = \left\{ \begin{array}{l} E(\rho_j,t) \\ T(\rho_j,t) \end{array} \right.$$

$$\mathbf{\Gamma}_h^k(\rho_j,t) = \left\{ \begin{array}{l} D_0 E(\rho_j,t) Q_E(\rho_j,t) \hat{\boldsymbol{\rho}} \\ \left[\frac{3}{2}\chi_0 T^{\frac{3}{2}}(\rho_j,t) q^{\frac{3}{2}} Q_T(\rho_j,t) - q_{\mathsf{h}}\big(E(\rho_j,t), T(\rho_j,t), Q_T(\rho_j,t)\big) \right] \hat{\boldsymbol{\rho}} \end{array} \right.$$

$$\mathbf{S}_h^k(\rho_j,t) = \left\{ \begin{array}{l} E(\rho_j,t)\gamma\big(T(\rho_j,t), Q_T(\rho_j,t)\big) - [\gamma_0 + \beta E(\rho_j,t)]E(\rho_j,t) \\ S(\rho_j) \end{array} \right.$$

In the implementation of the scheme, these results are inserted in Eq. (4.34).

Next step is to implement the boundary conditions, starting with the Neumann boundary conditions at the plasma center (4.8). Recalling the '−/+' terminology used in writing the Lax-Friedrichs flux, Eq. (4.21), the homogeneous Neumann implementation at the plasma center corresponds to

$$\mathbf{U}_h^- = \mathbf{U}_h^+, \qquad \mathbf{Q}_h^- = -\mathbf{Q}_h^+. \tag{4.37}$$

At the plasma edge, the vanishing turbulence flux in the radial direction, (4.9), is

$$\nabla \cdot \mathbf{\Gamma}_E^{\mathrm{1D}} = \frac{1}{\rho}\mathbf{\Gamma}_E^{\mathrm{1D}} + \frac{\partial \mathbf{\Gamma}_E^{\mathrm{1D}}}{\partial \rho} = 0, \tag{4.38}$$

which suggests that

$$\mathbf{\Gamma}^{+}_{h,1} = -\rho_{N_p}\left[\mathcal{D}^K\vec{\mathbf{\Gamma}}^K_{h,1}\Big|_{\rho_{N_p}}\right] \tag{4.39}$$

should be applied at the edge, i.e. at the last node, ρ_{N_p}, of the last element, K.

The remaining boundary condition is the inhomogeneous Dirichlet condition at the plasma edge, (4.10). It corresponds to

$$\mathbf{U}^{+}_{h} = -\mathbf{U}^{-}_{h} + 0.8\texttt{keV}, \qquad \mathbf{Q}^{-}_{h} = \mathbf{Q}^{+}_{h}. \tag{4.40}$$

The main advantages of a 2D finite element numerical solution to Eqs. (4.1-4.2) over a cylindrical coordinate grid approach, is the ability to model arbitrary reactor cross-section geometries. I.e. the concept of a geometrical plasma centre is abandoned. As a consequence, the Neumann requirement at $\rho = 0$, (4.8), is relaxed. The cylindrical coordinates are not appropriate anymore, so before moving on to 2D the TSTM will be translated to cartesian coordinates.

4.5 Translation to Cartesian Coordinates

Operators in the TSTM, Eqs. (4.1-4.2), that needs to be transformed are:

$$\begin{aligned} &\text{Radial derivative } \frac{\partial}{\partial\rho} && \text{(enters via } \gamma\text{)}, && (4.41)\\ &\text{gradient } \nabla && \text{(enters via } \mathbf{\Gamma}_E \text{ \& } \mathbf{\Gamma}_T\text{)}, && (4.42)\\ &\text{and divergence } \nabla\cdot && \text{(appears directly)}. && (4.43)\end{aligned}$$

Operators (4.42-4.43) are just represented in the cartesian forms, rather than the cylindrical forms, whereas operator (4.41) becomes

$$\begin{aligned} \frac{\partial}{\partial x}\rho &= \frac{\partial}{\partial x}\sqrt{x^2+y^2}\\ \frac{\partial\rho}{\partial x} &= \frac{1}{2}\frac{1}{\sqrt{x^2+y^2}}2x\\ \partial\rho &= \frac{x}{\rho(x,y)}\partial x\\ &\Downarrow\\ \frac{\partial}{\partial\rho} &= \frac{\rho(x,y)}{x}\frac{\partial}{\partial x}. \end{aligned} \tag{4.44}$$

If x is close to 0, the ∂y version of Eq. (4.44) is used instead;

$$\frac{\partial}{\partial \rho} = \frac{\rho(x,y)}{y}\frac{\partial}{\partial y}. \tag{4.45}$$

Regardless, the issue is minimised by choosing origo of the cartesian coordinate system in the middle of 4 grid points.

Cylindrical unit vectors are related to cartesian unit vectors by

$$\begin{align}
\hat{\boldsymbol{\rho}} &= \hat{\boldsymbol{x}}\cos\phi + \hat{\boldsymbol{y}}\sin\phi, \tag{4.46}\\
\hat{\boldsymbol{\phi}} &= \hat{\boldsymbol{y}}\cos\phi - \hat{\boldsymbol{x}}\sin\phi, \tag{4.47}\\
\hat{\boldsymbol{z}} &= \hat{\boldsymbol{z}}. \notag
\end{align}$$

Since

$$\begin{align}
x &= \rho\cos\phi, \tag{4.48}\\
y &= \rho\sin\phi, \tag{4.49}\\
z &= z, \tag{4.50}
\end{align}$$

it follows, that

$$\phi = \arctan\frac{y}{x}. \tag{4.51}$$

Eqs. (4.46-4.47) can then be expressed entirely in terms of cartesian coordinates;

$$\begin{align}
\hat{\boldsymbol{\rho}} &= \hat{\boldsymbol{x}}\cos\left(\arctan\frac{y}{x}\right) + \hat{\boldsymbol{y}}\sin\left(\arctan\frac{y}{x}\right), \tag{4.52}\\
\hat{\boldsymbol{\phi}} &= \hat{\boldsymbol{y}}\cos\left(\arctan\frac{y}{x}\right) - \hat{\boldsymbol{x}}\sin\left(\arctan\frac{y}{x}\right). \tag{4.53}
\end{align}$$

Thus, the TSTM, Eqs. (4.1-4.2), can be recast in cartesian coordinates. Using Eqs. (4.52-4.53), the 2D flux functions become:

$$\begin{align}
\boldsymbol{\Gamma}_E^{\text{2D}} &= D_0 E\nabla E\left[\hat{\boldsymbol{x}}\cos\left(\arctan\frac{y}{x}\right) + \hat{\boldsymbol{y}}\sin\left(\arctan\frac{y}{x}\right)\right] \notag\\
&\quad + [D_0 + D_{\text{pol}}(T)]E\nabla E\left[\hat{\boldsymbol{y}}\cos\left(\arctan\frac{y}{x}\right) - \hat{\boldsymbol{x}}\sin\left(\arctan\frac{y}{x}\right)\right] \tag{4.54}\\
\boldsymbol{\Gamma}_T^{\text{2D}} &= \left[\frac{3}{2}\chi_0 T^{\frac{5}{2}} q^{\frac{3}{2}}\frac{\nabla T}{T} - q_h\left(E, T, \frac{\partial T}{\partial \rho}\right)\right] \notag\\
&\quad \times \left[\hat{\boldsymbol{x}}\cos\left(\arctan\frac{y}{x}\right) + \hat{\boldsymbol{y}}\sin\left(\arctan\frac{y}{x}\right)\right] \notag\\
&\quad + \left[\frac{3}{2}\chi_0 T^{\frac{5}{2}} q^{\frac{3}{2}}\frac{\nabla T}{T} - q_h\left(E, T, \frac{\partial T}{\partial \rho}\right) + D_{\text{pol}}(T)\right] \notag\\
&\quad \times \left[\hat{\boldsymbol{y}}\cos\left(\arctan\frac{y}{x}\right) - \hat{\boldsymbol{x}}\sin\left(\arctan\frac{y}{x}\right)\right] \tag{4.55}
\end{align}$$

Keeping Eq. (4.44) in mind and using cartesian gradients and divergences, the 2D version of Eqs. (4.1-4.2) in cylindrical coordinates is thus translated to cartesian coordinates by replacing Eqs. (4.5-4.6) with Eqs. (4.54-4.55).

4.6 DG-FEM in 2 Dimensions

Inserting Eqs. (4.54-4.55) in Eqs. (4.1-4.2) yields the 2D TSTM in cartesian coordinates,

$$\begin{aligned}
\frac{\partial E}{\partial t} &= \nabla \cdot \Bigg[D_0 E \nabla E \left[\hat{\boldsymbol{x}} \cos\left(\arctan \frac{y}{x} \right) + \hat{\boldsymbol{y}} \sin\left(\arctan \frac{y}{x} \right) \right] \\
&\quad + [D_0 + D_{\text{pol}}(T)] E \nabla E \left[\hat{\boldsymbol{y}} \cos\left(\arctan \frac{y}{x} \right) - \hat{\boldsymbol{x}} \sin\left(\arctan \frac{y}{x} \right) \right] \Bigg] \\
&\quad + E\gamma \left(x, y, T, \frac{\partial T}{\partial \max(x,y)} \right) - [\gamma_0 + \beta E] E, \qquad (4.56) \\
\frac{\partial T}{\partial t} &= \nabla \cdot \Bigg[\left[\frac{3}{2} \chi_0 T^{\frac{5}{2}} q^{\frac{3}{2}} \frac{\nabla T}{T} - q_h \left(E, T, \frac{\partial T}{\partial \rho} \right) \right] \\
&\quad \times \left[\hat{\boldsymbol{x}} \cos\left(\arctan \frac{y}{x} \right) + \hat{\boldsymbol{y}} \sin\left(\arctan \frac{y}{x} \right) \right] \\
&\quad + \left[\frac{3}{2} \chi_0 T^{\frac{5}{2}} q^{\frac{3}{2}} \frac{\nabla T}{T} - q_h \left(E, T, \frac{\partial T}{\partial \rho} \right) + D_{\text{pol}}(T) \right] \\
&\quad \times \left[\hat{\boldsymbol{y}} \cos\left(\arctan \frac{y}{x} \right) - \hat{\boldsymbol{x}} \sin\left(\arctan \frac{y}{x} \right) \right] \Bigg] + S(x,y), \qquad (4.57)
\end{aligned}$$

with Eqs. (1.26-1.27) translated to

$$\gamma\left(x, y, T, \frac{\partial T}{\partial \max(x,y)}\right) = \lambda\sqrt{\frac{-R\frac{\sqrt{x^2+y^2}}{\max(x,y)}\frac{\partial T}{\partial \max(x,y)}}{T} - \kappa_c} \tag{4.58}$$

$$\text{for} \quad \frac{-R\frac{\sqrt{x^2+y^2}}{\max(x,y)}\frac{\partial T}{\partial \max(x,y)}}{T} > \kappa_c,$$

$$\gamma\left(x, y, T, \frac{\partial T}{\partial \max(x,y)}\right) = -\frac{\lambda}{10}\sqrt{\frac{R\frac{\sqrt{x^2+y^2}}{\max(x,y)}\frac{\partial T}{\partial \max(x,y)}}{T} + \kappa_c} \tag{4.59}$$

$$\text{for} \quad \frac{-R\frac{\sqrt{x^2+y^2}}{\max(x,y)}\frac{\partial T}{\partial \max(x,y)}}{T} < \kappa_c,$$

$$q_h\left(x, y, E, T, \frac{\partial T}{\partial \max(x,y)}\right) = CET\tanh\left(\gamma\left(x, y, T, \frac{\partial T}{\partial \max(x,y)}\right)\right). \tag{4.60}$$

Eqs. (4.56-4.57) are subject to the boundary conditions (4.9-4.10).

To cast Eqs. (4.56-4.57) in the DG-FEM weak or strong form, Eqs. (4.23) or (4.24), the solution, flux and source are identified as

$$\mathbf{U}_h^k(x_i, y_j, t) = \begin{cases} E_{ij} \\ T_{ij} \end{cases}$$

$$\mathbf{\Gamma}_h^k(x_i, y_j, t) = \begin{cases} \mathbf{\Gamma}_{ij}^{2\text{D}} \\ \mathbf{\Gamma}_{ij}^{2\text{D}} \end{cases}$$

$$\mathbf{S}_h^k(x_i, y_j, t) = \begin{cases} E\gamma\left(x_i, y_j, T_{ij}, \frac{\partial T}{\partial \max(x,y)}\right) - [\gamma_0 + \beta E_{ij}]E_{ij} \\ S_{ij} \end{cases}$$

respectively.

As the TSTM implementation developed here suggests, it would be preferable to start out with a model naturally formulated in cartesian coordinates, rather than Eqs. (4.1-4.2) given in cylindrical coordinates, for modeling the D-shaped 2D plasma profile observed in experiment.

Chapter 5

Conclusion

Within this chapter the findings presented in the former chapters are summarised. Then, the contributions made to the field of confined fusion research are highlighted, stressing the arguments and reasoning behind the work conducted. Finally, suggestions are made regarding future research within the field, based on ideas put forth in this thesis.

5.1 Summary of the Work Conducted

A numerical scheme accommodating the stiffness inherent in the CGM and TSTM has been developed. The diffusion equation was used to verify the numerical implementation of this TR-BDF2-based scheme, as this equation is analytically solvable and describes the underlying dynamics of the CGM and TSTM. Two distinct implementation strategies have been employed, namely the explicit- and direct derivatives approaches, and advantages and drawbacks have been identified for both.

Investigations based on the explicit derivatives approach include:

- Modeling of the TSTM as presented in [28].
- Modeling of the CGM as presented in [21].
- Modeling of the TSTM without gyroBohm scaling on the neo-classical diffusion term.

Reproduction of the TSTM results presented in [28] was not achieved. The results did not agree with expectations, namely stiffness of the region between plasma radial profile middle region was absent. No set of parameters was found to remedy this situation. Therefore a reference study was conducted, which indicated that the gyroBohm scaling on the neo-classical diffusion term is unphysical. In further support of removing the gyroBohm scaling, the CGM cold pulse modeling data that TSTM results are compared to in [28] was found to originate from [21], in which gyroBohm scaling is not included for the neo-classical term. Thus, numerical experiments was conducted with this CGM and good agreement was found with JET shot 55809 data, though linear growth in the plasma centre temperature time trace occurs at certain resolutions. Also, the equilibrium between the ρ-proportional heat flux and accumulated source was found to oscillate in the middle and edge regions were anomalous heat flux is present. Motivated by the findings of the reference study, the gyroBohm scaling and safety factor dependence was removed from the TSTM neo-classical diffusion term.

Investigations based on the direct derivatives approach include:

- Radial profile modeling of the CGM as presented in [28].
- Poloidal cross-section modeling using this CGM definition.

Agreement with experiment was found for the 1D implementation, and the cold pulse plasma core delay time was observed to be close to that stated in [28, 21]. Furthermore, simulation wall time was reduced considerably going from the explicit- to direct derivatives approach, and the deviation between ρ-proportional radial heat flux and accumulated source in steady-state was much reduced. Based on these results, the 1D implementation was assessed validated and the spatial region modeled was extended to include the poloidal plasma cross-section. Carefully verifying this 2D implementation against 1D results, it was ultimately used for poloidally asymmetric ICRH modeling, and interpretations of the results were offered.

Last, a numerical method suitable for modeling arbitrary plasma geometries with the CGM and TSTM has been presented.

5.2 Contributions to CGM and TSTM Research

As mentioned, the reference study revealed, that gyroBohm dependence on the neo-classical term is unphysical, and that assuming safety factor dependence is also questionable. Therefore, both were dropped from the TSTM definition [28]. The non-gyroBohm TSTM modeling results exhibited high sensitivity on the initial turbulent energy profile as well as on the model parameters: For simulations with high initial turbulent energy profile and small λ (< 1.3), the plasma centre temperature time trace was found to decrease (simulation end time at $2\texttt{s}$), whereas for high λ (> 1.7) the plasma centre temperature time trace was found to increase. For certain λ values in between, the temperature time trace appeared converged. It was found that high χ_0 values ($> 1\texttt{s}^{-1}$) are needed to stabilise simulations (otherwise the plasma centre temperature time trace do not approximately converge before cold pulse) for initial turbulent profiles above $0.001\texttt{keV}$. A comparison between heat modulation results for the non-gyroBohm CGM and TSTM was made, in which the CGM featured too fast heat wave propagation towards the plasma edge, whereas the TSTM featured too slow propagation, for the parameters modeled. Polarity reversal was observed at certain parameters, and the effect of including temperature gradient proportionality for the anomalous flux was found to amplify this pre-pulse fluctuation at the core. Fast cold pulse propagation was observed, though for parameter ratios far from those of [28].

TSTM results appearing in [28] for

- the turbulent energy and plasma temperature steady-states,

- the heat modulation amplitude and phase profiles,
- and cold pulse polarity reversal and fast propagation,

should be reproduced before any serious attempt of including the poloidal dynamics can be justified.

Upon validation of the 1D implementation of the CGM defined in [28], which is well-tested in terms of radial profile modeling [21, 9, 24, 22, 15], extension of the model was made in order to include dynamics of the fusion plasma poloidal cross-section. The 2D implementation was carefully constructed, step by step testing and comparing to 1D results, until credible 2D steady-state results were obtained using asymmetric ICRH, thus approximating the experimental conditions in JET shot 55809 in the poloidal plane. The 2D heat modulation results displayed an almost one-to-one correspondance between the radial 1D and 2D amplitude and phase profiles, at the 2D radial profile right between the outboard and inboard profiles ($\varphi = \frac{\pi}{2}$). I.e. no particular change in heat wave dynamics was observed as a consequence of including the poloidal dimension in the CGM simulation. The cold pulse, however, was observed to be damped, and even reversing sign of the thermal perturbation in the core for the outboard profile. The cold pulse launched at $\varphi = \frac{\pi}{2}$, i.e. between the outboard and inboard radial profiles, did not exhibit this change of sign in the core, though it was observed to be heavily damped. The interpretation offered for these observations is that the strong poloidal heat fluxes around the outboard profile — dictated by the asymmetric ICRH source localisation Gaussian — allow the warm pulses originating from the up-adjusted non cold pulse plasma edge grid points to engulf the outboard cold pulse in the core. Thereafter, due to the high poloidal heat fluxes in the core region, the time trace should approach the steady-state temperature there, as the poloidal average temperature is held fixed at the plasma edge. In the outboard core time trace, the temperature is indeed observed to drop steadily after the temperature rise. The time trace of the cold pulse launched from $\varphi = \frac{\pi}{2}$ could then be explained by the fact, that the poloidal heat flux is smaller in magnitude around this radial profile, and thus warm pulses originating from up-adjusted non cold pulse edge grid points would have less potential for engulfing the cold pulse.

An alternative to the cylindrically symmetric approach employed for modeling fusion plasma has been given, as the geometry of an actual fusion plasma is more D-shaped than round. The DG-FEM is assessed feasible to this end, as the CGM and TSTM can be interpreted as conservation laws. With this method the choice of spatial resolution for CGM and TSTM modeling is more flexible than for an FDM: Spatial resolution is controlled by the order of the interpolating Lagrange polynomial on each element, as well as the number of elements. Several advantages of the DG-FEM over the FEM and FVM are highlighted:

- Local mass matrix; low cost for performing spatial derivatives (which depends on the inverted mass matrix).
- Greater flexibility to ensure stability for wave dominated problems through carefully designing the numerical flux to reflect the underlying dynamics.
- High order accuracy on general grids, while maintaining local conservation and flexibility in the choice of numerical flux.

Backed by these arguments, the groundwork for 1D and 2D TSTM DG-FEM implementations has been laid out.

5.3 Suggestions for Future Research

This thesis has treated the description of thermal transport in a fusion plasma, using heuristic models based more on experience than on strong physics arguments. Consequently, the background theory consists of scientific papers, rather than textbook results, and often insufficient information is provided in order to reproduce results. Take for example the CGM definition implemented in section 3.5. It is based on [21] in which information of neo-classical diffusion, source profile, and boundary conditions used to obtain results, is missing. In general, this means that a considerable amount of time must be spend on either parameter fitting or communication with the author in question, in order to get up to speed with the present research frontline, before new contributions can be made. Therefore, focus has been on documenting the numerical implementations, parameters and initial- & boundary conditions used to generate the results in the work presented. If the information given in the main body of the thesis is found inadequate, please refer to the appendices for explicit implementation details.

As the results presented in this work indicate, the insight into the dynamics of a tokamak plasma offered by the models investigated here is limited. A few years back week-long computation times for nonlinear 3D fluid turbulence simulations over sufficient time intervals made such numerical tools unavailable to fusion researchers. Recently however, many advances have been made within high-performance scientific computing, gradually making computationally heavy $1st$ principle models more and more accessible. These rapid advances continually shortens the path towards numerical implementations of computationally demanding nonlinear gyro-kinetic models, derived from $1st$ principles (the Vlasov equation) and rich in detail. With regard to the heuristic CGM and TSTM studied in this work, a natural extension to the numerical investigations of the poloidal dynamics conducted would be to perform the corresponding study for a TSTM implementation that reproduces the results of [28]. Another option is to use the framework outlined in chapter 4 for future transport studies, however, the performance of the DG-FEM for fusion-related problems remains to be demonstrated. Pursuing the DG-FEM path, parallelisation of the developed code over spatial elements would allow high resolution investigations within reasonable time frames. With the promising potential of GPUs for parallel scientific computing, the cost of the necessary hardware drops further. A 3D model accommodating actual fusion plasma geometry could then be developed using the DG-FEM scheme, allowing more complete simulations of fusion plasma geometry and dynamics.

List of Tables

List of Figures

Appendix A

Analytical and Numerical Solution to Eq. (2.21)

A.1 Main Script

```
clear all
clc

global rho dr invdr inv2dr invdrsq splus

% compare against pseudo-analytical solution
% to inhomogeneous diffusion equation?
splus = 1;

% set timestep error
options=odeset('RelTol',1e-4,'AbsTol',1e-4,'Stats','on');

r_points = 1000;
% step size
dr = 1/r_points; invdr = 1/dr; invdrsq = invdr^2; inv2dr = 0.5*invdr;
% rho grid
rho = (dr/2:dr:r_points*dr)'; % ghost points included

% calculate profiles for times input below
tloopvec = 10;%[1E-3 1E-2 1E-1 5E-1 1 10];
```

```
for tloop = 1:length(tloopvec)
% construct straight line that is 0 at rho(end)
b = 10;
a = -b/rho(end);
U_initial = a*rho+b;%[a*rho+b a*rho+b];
U_initial = U_initial(:);
if splus, U_initial(:) = 0; end

% -------- ANALYTICAL SOLUTION TO HOMOGENEOUS DIFFFUISION EQUATION --------
terms = 100; % how many terms in sum over Bessel zeros
beszero = besselzero(0,terms,1)';
Cn = zeros(1,terms);
for Cnloop = 1:terms
    Cn(Cnloop) = 2./(rho(end)^2*besselj(1,beszero(Cnloop)).^2).*trapz(rho, ...
                 U_initial(:).*besselj(0,beszero(Cnloop)/rho(end)*rho).*rho);
             %U_initial(1:end/2).*besselj(0,beszero(Cnloop)/rho(end)*rho).*rho);
end
T_analytical = @(time,space) sum(Cn.*exp(-(beszero/rho(end)).^2.*time) ...
                                 .*besselj(0,beszero/rho(end)*rho(space)));
T_exact = zeros(1,length(rho));
for space = 1:length(rho)
    T_exact(space) = T_analytical(tloopvec(tloop),space);
end
% --- PSEUDO-ANALYTICAL SOLUTION TO INHOMOGENEOUS DIFFFUISION EQUATION ---
S = [0,0;0,-1];
optionsbvp = bvpset('SingularTerm',S);
solinit = bvpinit(linspace(0,1,length(rho)),@mat4init);
sol = bvp4c(@mat4ode,@mat4bc,solinit,optionsbvp);

% time integration interval
tspan = [0 tloopvec(tloop)];

tic

[t,U] = ode23tb(@rhs,tspan,U_initial,options);

toc

%  -----------------------------------------------------
% |                                                     |
% |          RESULT HANDLING BELOW THIS POINT           |
% |-----------------------------------------------------|
curv = ['b' 'g' 'm' 'r' 'c' 'k' 'y'];
leg{tloop} = sprintf('T=%0.3fs',tloopvec(tloop));
siz = size(U);
fig1 = figure(2);
hold on
subplot(1,3,1); plot(rho,T_exact,curv(tloop))
xlabel('\rho')
ylabel('''Exact'' value')
title('Analytical Solution')
%axis([0 rho(end) 0 10])
hold on
%subplot(1,3,2); plot(rho,U(end,1:end/2),curv(tloop))
subplot(1,3,2); plot(rho,U(end,:),curv(tloop))
```

```
xlabel('\rho')
ylabel('Numerical Solution')
title('Derivative in RHS function')
%axis([0 rho(end) 0 10])
hold on
subplot(1,3,3); plot(rho,abs(sol.y(1,:)-U(end,:))./sol.y(1,:),curv(tloop))
xlabel('\rho')
ylabel('Numerical Solution')
title('Derivative called')
%axis([0 rho(end) 0 10])
% subplot(1,3,3); plot(rho,U(end,end/2+1:end),curv(tloop))
% xlabel('\rho')
% ylabel('Numerical Solution')
% title('Derivative called')
% axis([0 rho(end) 0 10])
if tloop == length(tloopvec),legend(leg); end
exact(:,tloop) = T_exact';
first(:,tloop) = U(end,:)';
%secon(:,tloop) = U(end,end/2+1:end)';
end
```

A.2 RHS Function (rhs.m)

```
function [dU_dt] = rhs(t,U)

global rho stepcount invdr inv2dr invdrsq ghost splus

stepcount = stepcount+1;
fprintf('timestep #%u, timestep value = %f\n',stepcount,t)

Nr = length(U);%length(U)/2;
T  = U(1:Nr);
%T2 = U(Nr+1:end);

ghostT1 = T(1);
T_edge  = 0;
ghostT2 = 2*T_edge-T(Nr);

%ghost(1) = T2(2);
%T_edge  = 0;
%ghost(2) = 2*T_edge-T2(Nr-1);

% % step factor
% inv2r  = (2*h)^(-1);
% invhsq = h^(-2);

% 1st derivatives
dT = zeros(Nr,1);
dT(2:Nr-1)      = (T(3:Nr)-T(1:Nr-2))*inv2dr;
% handling of bounds
dT(1)           = (T(1)-ghostT1)*invdr;
```

```
dT(Nr)          = (ghostT2-T(Nr))*invdr;

% 2nd derivatives (edge ghost point not included in evaluations)
d2T = zeros(Nr,1);
d2T(2:Nr-1) = (T(3:Nr)+T(1:Nr-2)-2*T(2:Nr-1))*invdrsq;
% handling of bounds
d2T(1) = (T(2)+ghostT1-2*T(1))*invdrsq;
d2T(Nr) = (ghostT2+T(Nr-1)-2*T(Nr))*invdrsq;

dU_dt = zeros(length(U),1);

%  --------------------------------------------------
% |                                                  |
% |                  MODEL EQUATIONS                 |
% |--------------------------------------------------|

span = 1:Nr; % span over grid points to be evaluated
dU_dt(span)    = dT./rho+d2T;

if splus
    % source boost factor
    boost = 4.07;
    % amplitude fitting parameter
    ampfit = 0.21;
%     % source boost factor
%     boost = 1;
%     % amplitude fitting parameter
%     ampfit = 1;
    %--------------------SOURCE SETUP----------------------------
    NBI_HEATING = zeros(length(rho),1);                      %   |
    RF_HEATING  = zeros(length(rho),1);                      %   |
    source      = zeros(length(rho),1);                      %   |
    NBI_HEATING(:) = boost*0.6*(50*(1-rho(:)).^2+5);         %   |
    RF_HEATING(:)  = boost*70*0.6*exp(-(rho(:)-0.32).^2*17^2);%  |
    amp            = ampfit*0.75;                            %   |
    %-----------------END OF SOURCE SETUP------------------------
    source = NBI_HEATING+amp*RF_HEATING;
    span = 1:Nr-1; % span over grid points to be evaluated
    dU_dt(span) = dU_dt(span) + source(span);
end

%  -------------------------------------------------
% |                                                 |
% |            END OF EQUATION SYSTEM               |
% |-------------------------------------------------|

return
```

A.3 Help Scripts

These scripts aid the evaluation of the analytical and pseudo-analytical solutions to the homogeneous- and inhomogeneous diffusion equations, respectively.

A.3.1 Calculate Zeroes of the Bessel Function (besselzero.m)

```
function x=besselzero(n,k,kind)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% besselzero.m
%
% Find first k positive zeros of the Bessel function J(n,x) or Y(n,x)
% using Halley's method.
%
% Written by: Greg von Winckel - 01/25/05
% Contact: gregvw(at)chtm(dot)unm(dot)edu
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

k3=3*k;

x=zeros(k3,1);

for j=1:k3

    % Initial guess of zeros
    x0=1+sqrt(2)+(j-1)*pi+n+n^0.4;

    % Do Halley's method
    x(j)=findzero(n,x0,kind);

    if x(j)==inf
        error('Bad guess.');
    end

end

x=sort(x);
dx=[1;abs(diff(x))];
x=x(dx>1e-8);

x=x(1:k);

function x=findzero(n,x0,kind)

n1=n+1;     n2=n*n;
```

```
% Tolerance
tol=1e−12;

% Maximum number of times to iterate
MAXIT=100;

% Initial error
err=1;

iter=0;

while abs(err)>tol & iter<MAXIT

    switch kind
        case 1
            a=besselj(n,x0);
            b=besselj(n1,x0);
        case 2
            a=bessely(n,x0);
            b=bessely(n1,x0);
    end

    x02=x0*x0;

    err=2*a*x0*(n*a−b*x0)/(2*b*b*x02−a*b*x0*(4*n+1)+(n*n1+x02)*a*a);

    x=x0−err;
    x0=x;
    iter=iter+1;

end

if iter>MAXIT−1
    warning('Failed to converge to within tolerance. ',...
            'Try a different initial guess');
    x=inf;
end
```

A.3.2 Pseudo-analytical Solution Help Script (**mat4bc.m**)

```
function res = mat4bc(ya,yb)
res = [  ya(2)      % Neumann at rho = 0
         yb(1) ];   % Dirichlet at rho = R
end
```

A.3.3 Pseudo-analytical Solution Help Script (**mat4init.m**)

```
function yinit = mat4init(x)
% yinit = [ −32.7*x+32.7
```

```
%            -32.7 ];
      yinit = [ -134.31*x+134.31
          -134.31 ];
end
```

A.3.4 Pseudo-analytical Solution Help Script (**mat4ode.m**)

```
function dydx = mat4ode(x,y)
% source boost factor
boost = 4.07;
% amplitude fitting parameter
ampfit = 0.21;

% % source boost factor
% boost = 1;
% % amplitude fitting parameter
% ampfit = 1;

%--------------------SOURCE SETUP----------------------------
% NBI_HEATING = zeros(length(rho),1);                       %   |
% RF_HEATING  = zeros(length(rho),1);                       %   |
% SOURCE      = zeros(length(rho),1);                       %   |
% NBI_HEATING(:) = boost*0.6*(50*(1-rho(:)).^2+5);          %   |
% RF_HEATING(:)  = boost*70*0.6*exp(-(rho(:)-0.32).^2*17^2);%   |
amp            = ampfit*0.75;                             %   |
%-----------------END OF SOURCE SETUP------------------------
dydx = [ y(2)
         -(boost*0.6*(50*(1-x)^2+5)+amp*boost*70*0.6*exp(-(x-0.32)^2*17^2)) ];
end
```

Appendix B

Explicit Derivatives Implementation

B.1 Main Script

```
clear all
close all
clc

% Radial profile time—integration of the Critical Gradient Model (CGM)
% & the Turbulence Spreading Transport Model (TSTM)

% rho = [0,0.87] (0) or rho = [0,1] (1)
rhospan = 0;

% =======CHOOSE SIMULATION MODE=======
% ====================================
% 1 for STEADY STATE convergence
% 2 for SOURCE MODULATION
% 3 for COLD PULSE
sim_mode = 1;
% 4 for CGM
% 5 for TSTM
% 6 for both
% 7 for test
modelsim = 6;
```

```
% ====================================
% ====================================


% ======= SPECIFY TIME INPUT ===============
% calculate profiles for times input below
tloopvec = 1;%[1E-2 5E-2 3E-1 5E-1 2];
% set timestep error
options  = odeset('RelTol',1e-3,'AbsTol',1e-4,'Stats','on');
% ============== END =======================


% ======= SPECIFY RESOLUTION ===============
% set spatial resolution
grid_points = 100;
% ============== END =======================


% ======= CONSTRUCT RADIAL GRID ============
% step size
h = 0.87/(grid_points);
if rhospan, h = 1/(grid_points); end
% radial grid
rho = (h/2:h:(grid_points)*h)';
% ============== END =======================


% ======= SPECIFY SPATIAL DISCRETISATION ===
% 1) central approximation, 2nd order accurate
% 2) upwind scheme        , 3rd order accurate
% 3) 5-point stencil      , 6th order accurate
fdm = 1;
% (Only used for cold  pulse simulations as
%  no significant resolution advantages are
%  observed for 2)  and 3) for steady-state
%  and    heat    modulation    experiments)
% ============== END =======================


% ======= SPECIFY INITIAL CONDITIONS =======
% flat profile initial conditions (edge value for T)
ICturbEint = 1e-1;
DirichletT = 7.853e-1;
if rhospan, DirichletT = 0.4628; end
% ============== END =======================


% ======= SET PARAMETERS & LOAD DATA =======
% collect global help variables in array
helpvars = [sim_mode fdm];
% parameters = [ D0   h  lambda R gamma0 beta C kappac chi0 chiS q]
  parameters = [ 35  h     1.7  1   0.5   1  0.5  1.8   0.6   2 2];
% load heat modulation experimental data
[vartitle, moddata] = hdrload('./data/55809_exp_data.txt');
[oldtitle, olddata] = hdrload('./data/55809_CGM.txt');
```

```
% load plasma edge cold pulse Dirichlet data
load ./data/bd_cond.txt
% load experimental data->used later in spatial q, chi0, etc. investigation
[vars, expdat] = hdrload('./data/dati_chippy_new.txt');
% ------------- SOURCE SETUP ---------------
source = struct('NBI_HEATING',[],'RF_HEATING',[],     ...
                'amp',[],'amplitude',[],'boost',[],   ...
                'ICRHmodulation_t',[]);
source.NBI_HEATING = 2/3*(50*(1-rho).^2+5);
rhoICRH = 0.32;       %                   <-- ICRH local heating at rhoICRH
source.RF_HEATING  = 70*2/3*exp(-(rho-rhoICRH).^2*17^2);
% set asymmetrical heating param. and heat modulation amplitude & frequency
source.amp = 1.1; source.amplitude = 1; source.mod_freq = 15;
% set source boost fitting parameter
source.boost = 1 ; %(fit to article steady-state)
% start modulation of ICRH source at time:
source.ICRHmodulation_t = 0.7;
% ============== END =======================


% ======= COLLECT ALL DATA IN STRUCTURE ====
data = struct('rho',rho,'helpvars',helpvars,          ...
              'coldpulse',bd_cond,'edgeT',DirichletT, ...
              'source',source,'parameters',parameters);
% ============== END =======================

% use last input time for heat modulation or cold pulse simulations
if sim_mode ~= 1, tloopvec = tloopvec(end); end

for tloop = 1:length(tloopvec)

    if modelsim == 4
        CGM_initial = repmat(DirichletT,grid_points+2,1);
        % reshape to proper input in ode-solver
        CGM_initial = init;%CGM_initial(:);
        % time integration interval
        tspan = [0 tloopvec(tloop)];
        tic
        % solve CGM in 1 dimension
        [t,cgmT] = ode23tb(@(t,T) CGM_1D(t,T,data),tspan,CGM_initial,options);
        toc
        save cgmout rho t cgmT
    elseif modelsim == 5
        TSTM_initial = repmat([ICturbEint DirichletT],grid_points,1);
        %TSTM_initial = [ones(length(rho),1) linspace(2,DirichletT,length(rho))'];
        % reshape to proper input in ode-solver
        TSTM_initial = TSTM_initial(:);
        % time integration interval
        tspan = [0 tloopvec(tloop)];
        tic
        % solve TSTM in 1 dimension
        [t,TSTM] = ode23tb(@(t,U) TSTM_1D(t,U,data),tspan,TSTM_initial,options);
        toc
        Eout = TSTM(:,1:end/2); Tout = TSTM(:,end/2+1:end);
        chi0 = parameters(end-2); chiS = parameters(end-1);
```

```
        kappac = parameters(end-3); lambda = parameters(3);
        C = parameters(end-4);
        for tempo = 1:length(t)
%             if mod(1000,tempo)
%                 fprintf('writing data: %u /%\r',round(100*tempo/length(t)))
%             end
            E          = Eout(tempo,:);
            T          = Tout(tempo,:);
            ghostT(1) = T(2);
            bd_cond    = data.coldpulse;
            T_edge     = bd_cond(abs(bd_cond(:,1)-t(tempo))== ...
                        min(abs(bd_cond(:,1)-t(tempo))),2);
            ghostT(2) = 2*T_edge(1)-T(end-1);
            dT         = dTdr(T, 1,1,data,ghostT);
            d2T        = dTdr(T, 1,2,data,ghostT);
            dTinv      = dTdr(T,-1,1,data,ghostT);
            kappaT     = -dT./T;
            gam        = -0.1*lambda*sqrt(kappac-kappaT);
            dgam       = -0.1*lambda*(d2T./T+dT.*dTinv)./sqrt(kappac-kappaT);
            posgrowth = logical(imag(gam));
            gam(posgrowth) = lambda*sqrt(kappaT(posgrowth)-kappac);
            dgam(posgrowth) = -lambda*(d2T(posgrowth)./T(posgrowth)+dT(posgrowth) ...
                            .*dTinv(posgrowth))./sqrt(kappaT(posgrowth)-kappac);
            grad(tempo,:) = dT./T;
            anom(tempo,:) = -C*E.*T.*tanh(gam);
            neoc(tempo,:) = chi0*dT;
            flux(tempo,:) = anom(tempo,:) + neoc(tempo,:);
            fluxTnorm(tempo,:) = (anom(tempo,:) + neoc(tempo,:))./T;
        end
        cutupperr = find(t<1.08);
        CPindex = find(t(cutupperr)>1.01);
        rindex = [round(length(rho)*0.11/rho(end))  ...
                  round(length(rho)*0.17/rho(end))  ...
                  round(length(rho)*0.30/rho(end))  ...
                  round(length(rho)*0.45/rho(end))  ...
                  round(length(rho)*0.63/rho(end))  ...
                  round(length(rho)*0.51/rho(end))];
        fluxtrace = flux(CPindex,rindex);%fluxTnorm(CPindex,rindex);
        gradtrace = grad(CPindex,rindex);
% -------------------------------------------------------------------------
    elseif modelsim == 6
        CGM_initial = repmat(DirichletT,grid_points,1);
        % reshape to proper input in ode-solver
        CGM_initial = CGM_initial(:);
        % time integration interval
        tspan = [0 tloopvec(tloop)];
        tic
        % solve CGM in 1 dimension
        [t,cgmT] = ode23tb(@(t,T) CGM_1Dbeta08(t,T,data),tspan,CGM_initial,options);
        chi0 = parameters(end-2); chiS = parameters(end-1); kappac = parameters(end-3);
        if sim_mode == 2 % EXTRACT HEAT MODULATION FLUX-GRADIENT RELATIONSHIP
        for tempo = 1:length(t)
            T           = cgmT(tempo,:);
            T3h         = T.^(3/2);
            ghostT(1)  = T(2);
```

```
            DirichletT = data.edgeT;
            T_edge     = DirichletT;
            ghostT(2)  = 2*T_edge-T(end-1);
            dT         = dTdr(T, 1 ,1,data,ghostT);
            d2T        = dTdr(T, 1 ,2,data,ghostT);
            dT3h       = dTdr(T,3/2,1,data,ghostT);
            kappaT     = -dT./T;
            H          = heaviside(kappaT-kappac);
            grad(tempo,:) = dT./T;
            anom(tempo,:) = chiS*T3h.*(-dT./T-kappac).^(4/5).*H.*dT;
            neoc(tempo,:) = chi0*dT;
            flux(tempo,:) = chi0*dT+chiS*T3h.*(-dT./T-kappac).^(4/5).*H.*dT;
        end
        HMindex = find(t>1.5);
        rindex = [round(length(rho)*0.11/rho(end))  ...
                  round(length(rho)*0.17/rho(end))  ...
                  round(length(rho)*0.30/rho(end))  ...
                  round(length(rho)*0.45/rho(end))  ...
                  round(length(rho)*0.63/rho(end))  ...
                  round(length(rho)*0.78/rho(end))];
        fluxtrace = flux(HMindex,rindex);
        gradtrace = grad(HMindex,rindex);
        elseif sim_mode == 3 % EXTRACT COLD PULSE FLUX-GRADIENT RELATIONSHIP
        for tempo = 1:length(t)
            T = cgmT(tempo,:);
            T3h = T.^(3/2);
            ghostT(1) = T(2);
            bd_cond = data.coldpulse;
            T_edge  = bd_cond(abs(bd_cond(:,1)-t(tempo))==  ...
                              min(abs(bd_cond(:,1)-t(tempo))),2);
            ghostT(2) = 2*T_edge(1)-T(end-1);
            dT     = dTdr(T, 1 ,1,data,ghostT);
            d2T    = dTdr(T, 1 ,2,data,ghostT);
            dT3h   = dTdr(T,3/2,1,data,ghostT);
            kappaT = -dT./T;
            H      = heaviside(kappaT-kappac);
            grad(tempo,:) = dT./T;
            anom(tempo,:) = chiS*T3h.*(-dT./T-kappac).^(4/5).*H.*dT;
            neoc(tempo,:) = chi0*dT;
            flux(tempo,:) = chi0*dT+chiS*T3h.*(-dT./T-kappac).^(4/5).*H.*dT;
        end
        CPindex = find(t>1.005);
        rindex = [round(length(rho)*0.11/rho(end))  ...
                  round(length(rho)*0.17/rho(end))  ...
                  round(length(rho)*0.30/rho(end))  ...
                  round(length(rho)*0.45/rho(end))  ...
                  round(length(rho)*0.63/rho(end))  ...
                  round(length(rho)*0.78/rho(end))];
        fluxtrace = flux(CPindex,rindex);
        gradtrace = grad(CPindex,rindex);
        end
        toc
    elseif modelsim == 7
        testIC = -8.*rho+10;
        % reshape to proper input in ode-solver
```

```
        test_initial = testIC(:);
        % time integration interval
        tspan = [0 tloopvec(tloop)];
        tic
        % solve CGM in 1 dimension
        [t,test] = ode23tb(@test_1D,tspan,test_initial,options);
        toc
    else
        fprintf('wrong input...')
    end

end

% %  -----------------------------------------------------
% % |                                                     |
% % |          RESULT HANDLING BELOW THIS POINT           |
% % |-----------------------------------------------------|
```

B.2 CGM RHS Function (**CGM_1Dbeta08.m**)

```
function [dT_dt] = CGM_1Dbeta08(t,T,data)

persistent stepcount plotvar ttrace nonlintrace

global flux gradTtrace nonlinfluxes

if isempty(stepcount), stepcount = 0; plotvar = 0; ttrace = 0; nonlintrace = 0; end

parameters = data.parameters;

%D0     = parameters(1);
%h      = parameters(2);
%lambda = parameters(3);
%R      = parameters(4);
%gamma0 = parameters(5);
%beta   = parameters(6);
%C      = parameters(7);
kappac = parameters(8);
chi0   = parameters(9);
chiS   = parameters(10);
%q      = parameters(11);

helpvars = data.helpvars;

sim_mode = helpvars(1);
fdm      = helpvars(2);

stepcount = stepcount+1;
if mod(stepcount,10000) == 0
    fprintf('timestep #%u, timestep value = %f\r',stepcount,t)
    ttrace = round(ttrace + 1);
```

```
end

Nr = length(T); % No. grid points

% ---------- BOUNDARY CONDITIONS & COLD PULSE IMPLEMENTATION ----------

% PLASMA TEMPERATURE
ghostT(1) = T(1); % plasma center: Neumann BC (update ghost point)
if sim_mode ~= 3
     DirichletT = data.edgeT;
     T_edge   = DirichletT;
     ghostT(2) = 2*T_edge-T(end);
else
%   cold pulse data
    bd_cond = data.coldpulse;
    T_edge  = bd_cond(abs(bd_cond(:,1)-t)==min(abs(bd_cond(:,1)-t)),2);
    ghostT(2) = 2*T_edge(1)-T(end);
end

% -------------------------- END OF BC DEFINITIONS --------------------------

% load derivatives
dT    = dTdr(T,1,1,data,ghostT);  d2T  = dTdr(T,1,2,data,ghostT);
dT3h = dTdr(T,3/2,1,data,ghostT); dTinv = dTdr(T,-1,1,data,ghostT);

% switch stencil
if     t>1.01 && fdm ==2 && sim_mode == 3  % cold pulse

    % upwind stencil (3rd order accurate)
    dT    = dTdrUpwind(T,1,1,data,ghostT);
    d2T   = dTdrUpwind(T,1,2,data,ghostT);
    dT3h  = dTdrUpwind(T,3/2,1,data,ghostT);
    dTinv = dTdrUpwind(T,-1,1,data,ghostT);

elseif t>1.01 && fdm ==3 && sim_mode ~= 1  % heat modulation & cold pulse

    % 5-point stencil (6th order accurate)
    dT    = dTdr5point(T, 1 ,1,data,ghostT);
    d2T   = dTdr5point(T, 1 ,2,data,ghostT);
    dT3h  = dTdr5point(T,3/2,1,data,ghostT);
    dTinv = dTdr5point(T,-1 ,1,data,ghostT);
end


% load commonly used T exponential variables
T1h = T.^(1/2); T3h = T.^(3/2);

%  --------------------------------------------------
% |                                                   |
% |                  MODEL EQUATION                   |
% |--------------------------------------------------|

kappaT        = -dT./T;
H       = heaviside(kappaT-kappac); %0.5*(1+tanh((kappaT-kappac)));
```

```
rho     = data.rho;


grad = dT./T;
anom = chiS*T3h.*(-dT./T-kappac).^(4/5).*H.*dT;
neoc = chi0*dT;
flux = chi0*dT+chiS*T3h.*(-dT./T-kappac).^(4/5).*H.*dT;

% % ---------------------------------------------------------------------------
% invrhoterm  = 1./rho.*(T3h.*dT*chiS.*(kappaT-kappac).^(4/5).*H+chi0*dT);
% % ---------------------------------------------------------------------------
% stiffT3der  = chiS*H.*dT3h.*(kappaT-kappac).^(4/5);
% % ---------------------------------------------------------------------------
% kappaderiv  = chiS*T3h.*H.*dT*4/5.*(kappaT-kappac).^(-1/5).*(-d2T./T-dTinv.*dT);
% % ---------------------------------------------------------------------------
% doublediff  = T3h.*d2T*chiS.*(kappaT-kappac).^(4/5).*H;
% % ---------------------------------------------------------------------------
% neoclasdiff = chi0*d2T;
% % ---------------------------------------------------------------------------
% ---------------------------------------------------------------------------
invrhoterm  = 1./rho.*(chi0*dT+chiS*T3h.*dT.*(kappaT-kappac).^(4/5).*H);
% ---------------------------------------------------------------------------
stiffT3der  = chiS*H.*dT3h.*(kappaT-kappac).^(4/5).*dT;
% ---------------------------------------------------------------------------
kappaderiv  = chiS*T3h.*H*4/5.*(kappaT-kappac).^(-1/5).*(-d2T./T-dTinv.*dT).*dT;
% ---------------------------------------------------------------------------
doublediff  = chiS*T3h.*(kappaT-kappac).^(4/5).*H.*d2T;
% ---------------------------------------------------------------------------
neoclasdiff = chi0*d2T;
% ---------------------------------------------------------------------------

dT_dt = invrhoterm + stiffT3der + kappaderiv + doublediff + neoclasdiff;

% ---------------------------------------------------------------------------

% load source profile
NBI   = data.source.NBI_HEATING;
RF    = data.source.RF_HEATING;
amp   = data.source.amp;
boost = data.source.boost;
mod_t = data.source.ICRHmodulation_t;

% modulation heat wave
% if sim_mode == 2 && t>0.7
%     pulse_factor = t*data.source.mod_freq;
%     fpulse_factor = floor(pulse_factor);
%     amp = 0.5 + heaviside((pulse_factor-fpulse_factor)*2-1);
% end
% modulation heat wave
if sim_mode == 2 && t>mod_t
    amplitude     = data.source.amplitude;
    P_mod_off     = amp-amplitude; % switch off asymmetrical heating
    P_mod_on      = 2*amplitude;   % switch on  asymmetrical heating
    pulse_factor  = t*data.source.mod_freq;
    fpulse_factor = floor(pulse_factor);
```

```
    onORoff       = pulse_factor-fpulse_factor;
    amp           = P_mod_off + heaviside(onORoff*2-1)*P_mod_on;
end
% if sim_mode == 2 && t>0.7
%     P_mod_off     = 0; % switch off asymmetrical heating
%     P_mod_on      = data.source.amp;   % switch on  asymmetrical heating
%     pulse_factor  = t*data.source.mod_freq;
%     fpulse_factor = floor(pulse_factor);
%     onORoff       = pulse_factor-fpulse_factor;
%     amp           = P_mod_off + heaviside(onORoff*2-1)*P_mod_on;
% end


S = NBI+amp*RF;

% add source
span = 1:Nr-1; % no source on edge (no material to heat)
               % ((and undermines T edge BC definition))

dT_dt(span) = dT_dt(span)+boost*S(span);%(boost*(rho(end)-rho(span))+1).*S(span);

%  -------------------------------------------------
% |                                                 |
% |                 END OF EQUATION                 |
% |-------------------------------------------------|

plotvar = plotvar+1;

sourceintegral = cumtrapz(rho,rho.*S);

return
```

B.3 TSTM RHS Function (TSTM_1D.m)

```
function [dU_dt] = TSTM_1D(t,U,data)

global grad gradT60 tbflux anom neoc flux60

persistent stepcount plotvar nonlintrace

if isempty(stepcount), stepcount = 0; plotvar = 0; nonlintrace = 0; end

% U = [E T]

parameters = data.parameters;

D0     = parameters(1);
%h      = parameters(2);
lambda = parameters(3);
R      = parameters(4);
gamma0 = parameters(5);
```

```
beta   = parameters(6);
C      = parameters(7);
kappac = parameters(8);
chi0   = parameters(9);
%chiS   = parameters(10);
q      = parameters(11);

rho = data.rho;

helpvars = data.helpvars;

sim_mode = helpvars(1);
fdm      = helpvars(2);

stepcount = stepcount+1;
if mod(stepcount,1000) == 0
    fprintf('timestep #%u, timestep value = %f\r',stepcount,t)
end

% No. grid points
Nr = length(U)/2;

% extract E and T variables
E = U(1:Nr);
T = U(Nr+(1:Nr));

% -------- BOUNDARY CONDITIONS WITH COLD PULSE IMPLEMENTATION --------

% TURBULENCE ENERGY
ghostE(1) = E(1); % plasma center: Neumann BC (update ghost point)
% turbulence spreading term required to vanish at edge
ghostE(2) = E(end);%-0.5*(2*h-R*E(Nr-1)+4*E(Nr)*R-sqrt(4*h^2+4*h*R*E(Nr-1)+16*h ...
            %*E(Nr)*R-3*R^2*E(Nr-1)^2-24*R^2*E(Nr-1)*E(Nr)+48*E(Nr)^2*R^2))/R;

% PLASMA TEMPERATURE
ghostT(1) = T(2); % plasma center: Neumann BC (update ghost point)
if sim_mode ~= 3
     DirichletT = data.edgeT;
     T_edge  = DirichletT;
     ghostT(2) = 2*T_edge-T(end-1);
else
%   cold pulse data
    bd_cond = data.coldpulse;
    T_edge  = bd_cond(abs(bd_cond(:,1)-t)==min(abs(bd_cond(:,1)-t)),2);
    ghostT(2) = 2*T_edge(1)-T(end-1);
end

% ---------------------- END OF BC DEFINITIONS ----------------------

% load derivatives
dE    = dEdr(E,1,1,data,ghostE);  d2E  = dEdr(E,1,2,data,ghostE);
dT    = dTdr(T,1,1,data,ghostT);  d2T  = dTdr(T,1,2,data,ghostT);
dTinv = dTdr(T,-1,1,data,ghostT); dT3h = dTdr(T,3/2,1,data,ghostT);

% switch stencil
```

```
if     t>1.01 && fdm ==2 && sim_mode == 3 % cold pulse

    % upwind stencil (3rd order accurate)
    dE    = dEdrUpwind(E, 1 ,data,ghostE);
    d2E   = dEdrUpwind(E, 2 ,data,ghostE);
    dT    = dTdrUpwind(T, 1 ,1,data,ghostT);
    d2T   = dTdrUpwind(T, 1 ,2,data,ghostT);
    dTinv = dTdrUpwind(T,-1 ,1,data,ghostT);
    dT3h  = dTdrUpwind(T,3/2,1,data,ghostT);

elseif t>1.01 && fdm ==3 && sim_mode ~= 1 % heat modulation & cold pulse

    % 5-point stencil (6th order accurate)
    dE    = dEdr5point(E, 1 ,1,data,ghostE);
    d2E   = dEdr5point(E, 1 ,2,data,ghostE);
    dT    = dTdr5point(T, 1 ,1,data,ghostT);
    d2T   = dTdr5point(T, 1 ,2,data,ghostT);
    dTinv = dTdr5point(T,-1 ,1,data,ghostT);
    dT3h  = dTdr5point(T,3/2,1,data,ghostT);
end

% calculate gamma and its derivative
kappaT = -R*dT./T;
gam    = -0.1*lambda*sqrt(kappac-kappaT);
dgam   = -0.1*lambda*R*(d2T./T+dT.*dTinv)./sqrt(kappac-kappaT);
posgrowth = logical(imag(gam));
%gam(:) = 0; dgam(:) = 0;
gam(posgrowth) = lambda*sqrt(kappaT(posgrowth)-kappac);

% % ----- discontinuity straight line interpolation -----
% clearvars turbregion strlindex intpolslope dummygrowth
% dummygrowth = posgrowth;
% dummygrowth(1:20) = 0; dummygrowth(end-7:end) = 0;
% turbregion = find(dummygrowth);
% if any(dummygrowth)
%     strlindex = round(turbregion(1)-8:turbregion(1)+7);
%     intpolslope = (gam(strlindex(end))-gam(strlindex(1))) ...
%                   /(length(strlindex)-1);
%     gam(strlindex) = intpolslope*(strlindex'-strlindex(1))+gam(strlindex(1));
% end
% % -----------------------------------------------------

dgam(posgrowth) = -lambda*R*(d2T(posgrowth)./T(posgrowth)+dT(posgrowth) ...
                            .*dTinv(posgrowth))./sqrt(kappaT(posgrowth)-kappac);
% loft = find(dgam>100); gulv = find(dgam<100);
% if any(loft), dgam(loft) = 0; end
% if any(gulv), dgam(gulv) = 0; end

% % ----- discontinuity straight line interpolation -----
% clearvars turbregion strlindex intpolslope
% posgrowth(end-7:end) = 0;
% turbregion = find(posgrowth);
% if any(posgrowth)
%     strlindex = round(turbregion(1)-8:turbregion(1)+7);
%     intpolslope = (dgam(strlindex(end))-dgam(strlindex(1))) ...
```

```
%                    /(length(strlindex)-1);
%     dgam(strlindex) = intpolslope*(strlindex'-strlindex(1))+dgam(strlindex(1));
% end
% % -----------------------------------------------------

% ensure correct dU_dt size and dimension (otherwise dU_dt will miss last T point
% and also, when fed back into the ode solver, dU_dt will have wrong dimension )
dU_dt = zeros(length(U),1);

%   --------------------------------------------------
% |                                                  |
% |                  MODEL EQUATIONS                 |
% |--------------------------------------------------|

            anom = -C*E.*T.*tanh(gam);
            neoc = 3/2*q^(3/2)*chi0*T.^(3/2).*dT;%chi0*dT;
            tbflux = anom + neoc;

% ------------------------------------------------------------------------
spread = D0*(E.*dE./rho + dE.*dE + E.*d2E);
% ------------------------------------------------------------------------
divtbs = -C*(E.*T.*tanh(gam)./rho + tanh(gam).*(dE.*T+E.*dT) ...
         + E.*T.*dgam.*(1-tanh(gam).^2));
%  divtbs = -C*dT.*(E.*T.*tanh(gam)./rho + tanh(gam).*(dE.*T+E.*dT) ...
%           + E.*T.*dgam.*(1-tanh(gam).^2)) ...
%           -C*E.*T.*tanh(gam).*d2T;
% ------------------------------------------------------------------------
divneo = 3/2*q*sqrt(q)*chi0*((dT./rho + d2T).*T.^(3/2) + dT3h.*dT);
%divneo = chi0*(dT./rho + d2T);
% ------------------------------------------------------------------------

span = 1:Nr;

%dU_dt(span)    = 2*(spread+gam.*E-(gamma0+beta*E).*E);
dU_dt(span)    = spread+gam.*E-(gamma0+beta*E).*E;
dU_dt(Nr+span) = (divtbs + divneo);
%dU_dt(Nr+span) = divtbs + divneo;

% load source profile
NBI   = data.source.NBI_HEATING;
RF    = data.source.RF_HEATING;
amp   = data.source.amp;
boost = data.source.boost;
mod_t = data.source.ICRHmodulation_t;

% modulation heat wave
if sim_mode == 2 && t>mod_t
    amplitude     = data.source.amplitude;
    P_mod_off     = amp-amplitude; % switch off asymmetrical heating
    P_mod_on      = 2*amplitude;   % switch on  asymmetrical heating
    pulse_factor  = t*data.source.mod_freq;
    fpulse_factor = floor(pulse_factor);
    onORoff       = pulse_factor-fpulse_factor;
    amp           = P_mod_off + heaviside(onORoff*2-1)*P_mod_on;
end
```

```
S = NBI+amp*RF;

% add source
span = 1:Nr-1; % no source on edge (no material to heat)
               % ((and undermines T edge BC definition))

dU_dt(Nr+span) = dU_dt(Nr+span)+S(span);

%  --------------------------------------------------
% |                                                  |
% |             END OF EQUATION SYSTEM               |
% |--------------------------------------------------|

plotvar = plotvar+1;

if ~isreal(gam)

    stop
end

return
```

B.4 Spatial Stencils

B.4.1 Central approximation of E, Eqs. (2.6-2.7)

```
function [out] = dEdr(in,expon,order,data,ghostE)

h = data.parameters(2);
out = zeros(size(in));

% 3-point stencil
if order == 1
    out(2:end-1) = (in(3:end).^expon-in(1:end-2).^expon)/(2*h);
    out(1)       = (in(2)^expon-ghostE(1)^expon)/(2*h);
    out(end)     = (ghostE(2)^expon-in(end-1)^expon)/(2*h);
elseif order == 2
    out(2:end-1) = (in(3:end)+in(1:end-2)-2*in(2:end-1))/h^2;
    out(1)       = (in(2)+ghostE(1)-2*in(1))/h^2;
    out(end)     = (ghostE(2)+in(end-1)-2*in(end))/h^2;
else
    fprint('wrong arguments...')
end
```

B.4.2 Central approximation of T, Eqs. (2.6-2.7)

```
function [out] = dTdr(in,expon,order,data,ghostT)

Texp = in.^expon;

h = data.parameters(2);
out = zeros(size(in));

% 3-point stencil
if order == 1
    out(2:end-1) = (Texp(3:end)-Texp(1:end-2))/(2*h);
    out(1)       = (Texp(2)-ghostT(1)^expon)/(2*h);
    out(end)     = (ghostT(2)^expon-Texp(end-1))/(2*h);
elseif order == 2
    out(2:end-1) = (Texp(3:end)+Texp(1:end-2)-2*Texp(2:end-1))/h^2;
    out(1)       = (Texp(2)+ghostT(1)-2*Texp(1))/h^2;
    out(end)     = (ghostT(2)+Texp(end-1)-2*Texp(end))/h^2;
else
    fprint('wrong arguments...')
end
```

B.4.3 5-point approximation of E, Eqs. (2.9-2.10)

```
function [out] = dEdr5point(in,order,data,ghostE)

h = data.parameters(2);
out = zeros(size(in));

% 5-point stencil
if order == 1
    out(3:end-2) = (-in(5:end)+8*in(4:end-1) ...
                    -8*in(2:end-3)+in(1:end-4))/(12*h);
    out(2)       = (-in(4)+8*in(3)-8*in(1)+ghostE(1))/(12*h);
    out(end-1)   = (-ghostE(2)+8*in(end)-8*in(end-2)+in(end-3))/(12*h);
    % central approximation at boundary evaluation
    out(1)       = (in(2)-ghostE(1))/(2*h);
    out(end)     = (ghostE(2)-in(end-1))/(2*h);
elseif order == 2
    out(3:end-2) = (-in(5:end)+16*in(4:end-1)-30*in(3:end-2) ...
                    +16*in(2:end-3)-in(1:end-4))/(12*h^2);
    out(2)       = (-in(4)+16*in(3)-30*in(2)                   ...
                    +16*in(1)-ghostE(1))/(12*h^2);
    out(end-1)   = (-ghostE(2)+16*in(end)-30*in(end-1)           ...
                    +16*in(end-2)-in(end-3))/(12*h^2);
    % central approximation at boundary evaluation
    out(1)       = (in(2)+ghostE(1)-2*in(1))/(h^2);
    out(end)     = (in(end-1)+ghostE(2)-2*in(end))/(h^2);
else
    fprint('wrong arguments...')
end
```

B.4.4 5-point approximation for T, Eqs. (2.9-2.10)

```
function [out] = dTdr5point(in,expon,order,data,ghostT)

Texp = in.^expon;

h = data.parameters(2);
out = zeros(size(in));

% 5-point stencil
if order == 1
    out(3:end-2) = (-Texp(5:end)+8*Texp(4:end-1) ...
                    -8*Texp(2:end-3)+Texp(1:end-4))/(12*h);
    out(2)       = (-Texp(4)+8*Texp(3)-8*Texp(1)+ghostT(1)^expon)/(12*h);
    out(end-1)   = (-ghostT(2)^expon+8*Texp(end)-8*Texp(end-2)+ ...
                    Texp(end-3))/(12*h);
    % central approximation at boundary evaluation
    out(1)       = (Texp(2)-ghostT(1)^expon)/(2*h);
    out(end)     = (ghostT(2)^expon-Texp(end-1))/(2*h);
elseif order == 2
    out(3:end-2) = (-Texp(5:end)+16*Texp(4:end-1)-30*Texp(3:end-2) ...
                    +16*Texp(2:end-3)-Texp(1:end-4))/(12*h^2);
    out(2)       = (-Texp(4)+16*Texp(3)-30*Texp(2)                    ...
                    +16*Texp(1)-ghostT(1)^expon)/(12*h^2);
    out(end-1)   = (-ghostT(2)^expon+16*Texp(end)-30*Texp(end-1)
...
                    +16*Texp(end-2)-Texp(end-3))/(12*h^2);
    % central approximation at boundary evaluation
    out(1)       = (Texp(2)+ghostT(1)^expon-2*Texp(1))/(h^2);
    out(end)     = (Texp(end-1)+ghostT(2)^expon-2*Texp(end))/(h^2);
else
    fprint('wrong arguments...')
end
```

B.4.5 Upwind approximation of $\frac{\partial T}{\partial \rho}$, Eq. (2.8)

```
function [out] = dTdrUpwind(in,expon,order,data,ghostT)

Texp = in.^expon;

h = data.parameters(2);
out = zeros(size(in));

% 2nd order upwind
if order == 1
    out(2:end-2) = (-Texp(4:end)+6*Texp(3:end-1)-3*Texp(2:end-2) ...
                    -2*Texp(1:end-3))/(6*h);
    out(1)       = (Texp(2)-ghostT(1)^expon)/(2*h); % central approximation
    out(end-1)   = (Texp(end)-Texp(end-2))/(2*h);
    out(end)     = (ghostT(2)^expon-Texp(end-1))/(2*h);
```

```
elseif order == 2
    out(2:end-1) = (Texp(3:end)+Texp(1:end-2)-2*Texp(2:end-1))/h^2;
    out(1)       = (Texp(2)+ghostT(1)-2*Texp(1))/h^2;
    out(end)     = (ghostT(2)+Texp(end-1)-2*Texp(end))/h^2;
else
    fprint('wrong arguments...')
end
```

B.5 Heat Modulation Analysis

Essentially the same as in appendix D.4.

B.6 Cold Pulse Analysis

Essentially the same as in appendix D.5.

Appendix C

Direct Derivatives Implementation

C.1 Main Script

```
clear all; close all; clc

global S gam T chi qT32 dT_dt log_grad anom_flux neoc_flux total_flux

% Radial profile time-integration of the Critical Gradient Model (CGM)
% & the Turbulence Spreading Transport Model (TSTM)


% =======CHOOSE SIMULATION MODE=======
% ====================================
% 1 for STEADY-STATE
% 2 for SOURCE MODULATION
% 3 for COLD PULSE
sim_mode = 1;
% ----------------------------------
% 4 for CGM
% 5 for TSTM
modelsim = 4;
% ====================================
% ====================================
```

```
% ======= SPECIFY END TIME =================
% calculate profiles for times input below
end_time = 1;%[1E-2 5E-2 3E-1 5E-1 2];
% ============== END =======================


% ======= SPECIFY RESOLUTION ===============
% set spatial resolution
grid_points = 70;
% set timestep tolerances
options  = odeset('RelTol',1e-8,'AbsTol',1e-8,'Stats','on');
% ============== END =======================


% ======= SPECIFY INITIAL CONDITIONS =======
% flat profile initial conditions (edge value for T)
IC_E = 0.1;
IC_T = 7.853e-1;
% ============== END =======================


% ======= SPECIFY EQUATION PARAMETERS ======
% C         : connects heat flux with growth rate
% chi0      : neo-classical (collisional) diffusion
% lambda    : growth rate scaling
% crit_grad : critical gradient
% D0        : turbulence diffusion
% gam0      : turbulence damping
% beta      : turbulence nonlinear (quadratic) saturation
C  = 0.5; chi0 = 0.6; lambda = 1.7; crit_grad = 1.8;
D0 =  35; gam0 = 0.5; beta   =   1;
% ============== END =======================


% ======= RUN SETUP SCRIPT =================
initialise;
% ============== END =======================


% ======= SAFETY FACTOR PROFILE ============
q32 = (1 + r + 6*r.^4).^(3/2);
% ============== END =======================


% ======= COLLECT ALL INPUT IN STRUCTURE====
 input = struct('r',r,'invdr',invdr,'inv2dr',inv2dr,     ...
                'source',source,'parameters',parameters, ...
                'q32',q32,'edgeT',IC_T,'CGM_E',IC_E,     ...
                'modelsim',modelsim,'sim_mode',sim_mode);%, ...
                %'coldpulse',bd_cond);
                %'cp_edgedata',bd_cond,'edgeT',DirichletT, ...
% ============== END =======================

tic
```

```
[t,out] = ode23tb(@(t,ET) models1D(t,ET,input),[0 end_time],ICs,options);
toc

if modelsim == 4
    rho = r; T = out;
else
    rho = r; E = out(:,1:length(r)); T = out(:,length(r)+1:2*length(r));
end
%
% T1D = T(end,2:end-1);
% loggradr1D = log_grad(2:end-1);
% fluxr1D = total_flux;

T1D = T(end,:)';

save T1D

%writedata = [rho -log_grad(2:end-1)];
%save('loggradfluxupgradient.txt','writedata','-ASCII')
```

C.2 Initialisation Script (initialise.m)

```
% ======= CONSTRUCT RADIAL GRID ============
% step size
dr = 0.87/grid_points; invdr = 1./dr; inv2dr = 0.5*invdr;
% rho grid
r = (dr/2:dr:grid_points*dr)'; % ghost points included
% ============== END =======================


% ======= SOURCE SETUP =====================
source = struct('NBI_HEATING',[],'RF_HEATING',[],     ...
                'amp',[],'amplitude',[],'boost',[],   ...
                'ICRHmodulation_t',[]);
source.NBI_HEATING = 2/3*(50*(1-r).^2+5);
rhoICRH = 0.32;       %                  <-- ICRH local heating at rhoICRH
source.RF_HEATING  = 70*2/3*exp(-(r-rhoICRH).^2*17^2);
% set asymmetrical heating param. and heat modulation amplitude & frequency
source.amp = 0.75; source.amplitude = 0.55; source.mod_freq = 15;
% start modulation of ICRH source at time:
source.ICRHmodulation_t = 0.7;
% ============== END =======================


% % ======= LOAD DATA ========================
% load heat modulation experimental data
%[vartitle, moddata] = hdrload('./data/55809_exp_data.txt');
% plasma edge cold pulse Dirichlet data
%load ./data/bd_cond.txt
% load experimental data->used later in spatial q, chi0, etc. investigation
% [vars, expdat] = hdrload('./data/dati_chippy_new.txt');
```

```
% % load heat modulation experimental & CGM data
% [vartitle, moddata] = hdrload('./data/55809_exp_data.txt');
% [oldtitle, olddata] = hdrload('./data/55809_CGM.txt');
% % ============== END =======================


% ======= EQUATION PARAMETER ARRAY =========
parameters = [C chi0 lambda crit_grad D0 gam0 beta];
% ============== END =======================


% ======= SETUP INITIAL CONDITIONS =========
if     modelsim == 4 % CGM
    ICr = (-dr/2:dr:(grid_points+1)*dr)'; % r with ghost points included

%    ICs              = zeros(length(ICr),1);
%    ICs(:)           = IC_T*exp(crit_grad*(ICr(end)-ICr(:)));
    ICs              = zeros(size(r));
    ICs(:)           = IC_T*exp(crit_grad*(r(end)-r));
elseif modelsim == 5 % TSTM
    ICs              = zeros(2*length(r),1);
    ICs(1:end/2)     = IC_E;
    ICs(end/2+1:end) = IC_T*exp(crit_grad*(r(end)-r));
else
    disp('wrong modelsim input: choose ''4'' or ''5''');
    break
end
% ============== END =======================
```

C.3 CGM RHS Function (models1D.m)

```
function [dET_dt] = models1D(t,ET,input)

global S qT32 T gam chi dT_dt log_grad anom_flux neoc_flux total_flux

persistent stepcount plotvar

if isempty(stepcount), stepcount = 0; plotvar = 0; end

stepcount = stepcount+1;
if mod(stepcount,1000) == 0
    fprintf('timestep #%u, timestep value = %f\r',stepcount,t)
end

sim_mode = input.sim_mode;

r = input.r;
inv2dr = input.inv2dr;

C         = input.parameters(1);
chi0      = input.parameters(2);
```

```
lambda    = input.parameters(3);
crit_grad = input.parameters(4);

q32 = input.q32;

N = length(r)+2; % include ghost points
T = zeros(N,1);

no_g = 2:N−1; % index without ghost points
T(no_g) = ET;

T(1) = T(2);                 %   homogeneous Neumann   at r = 0
T(N) = 2*input.edgeT−T(N−1); % inhomogeneous Dirichlet at r = 0.87



qT32 = T(no_g).*sqrt(T(no_g));
q32mat = repmat(q32,1,1);
qT32 = q32mat.*qT32;
chi_collisional = chi0*qT32;

log_grad = zeros(N,1); % need ghost points
log_grad(no_g) = inv2dr*log(T(3:N)./T(1:N−2));

%gamma           = lambda*max(−log_grad(no_g)−crit_grad,−0.05); % upgradient
gamma           = lambda*max(−log_grad(no_g)−crit_grad,0.0); % Heaviside
gam             = sqrt(gamma);
posgrowth       = logical(imag(sqrt(gamma)));
gam(posgrowth) = −sqrt(−gamma(posgrowth));

chi = zeros(N,1); % needs ghost points

chi(no_g) = C*gam.*qT32;
anom_flux = chi(no_g).*T(no_g).*log_grad(no_g);

neoc_flux = chi_collisional.*T(no_g).*log_grad(no_g);
total_flux = anom_flux + neoc_flux;
chi(no_g) = chi(no_g) + chi_collisional; % total chi

FH = 2/3*0.372;

chi(1) = chi(2); chi(N) = chi(N−1); % chi has homogeneous Neumann at boundaries
log_grad(1) = log_grad(2);     % log_grad has homogeneous Neumann at r = 0
log_grad(N) = 2*log_grad(N−1) − log_grad(N−2); % slope continues at r = 0.87

invrmat = repmat(1./r,1,1);

dT_dt = FH*(invrmat.*T(no_g).*chi(no_g).*log_grad(no_g) ...
            + inv2dr*(T(3:N)−T(1:N−2)).*chi(no_g).*log_grad(no_g) ...
            + T(no_g).*inv2dr.*(chi(3:N)−chi(1:N−2)).*log_grad(no_g) ...
            + T(no_g).*chi(no_g).*inv2dr.*(log_grad(3:N)−log_grad(1:N−2)));


if any(imag(gam))
    gam;
```

```
    disp('Imaginary stuff')
    return
end

% load source profile
NBI   = input.source.NBI_HEATING;
RF    = input.source.RF_HEATING;
amp   = input.source.amp;
mod_t = input.source.ICRHmodulation_t;
mod_f = input.source.mod_freq;

if sim_mode == 2 && t>mod_t
    amplitude     = input.source.amplitude;
    P_mod_off     = amp-amplitude; % switch off asymmetrical heating
    P_mod_on      = 2*amplitude;   % switch on  asymmetrical heating
    pulse_factor  = t*mod_f;
    fpulse_factor = floor(pulse_factor);
    onORoff       = pulse_factor-fpulse_factor;
    amp           = P_mod_off + heaviside(onORoff*2-1)*P_mod_on;
end

S = NBI+amp*RF;

% add source
span = 1:length(r)-1; % no source on edge (no material to heat)
               % ((and undermines T edge BC definition))

dT_dt(span) = dT_dt(span)+S(span);
dET_dt = dT_dt;
```

C.4 Heat Modulation Analysis

Essentially the same as in appendix D.4.

C.5 Cold Pulse Analysis

Essentially the same as in appendix D.5.

Appendix D

Poloidal Cross-section Code

D.1 Main Script

```
clear all;
close all;
clc

global S gam_r qT32 chi_r T addpol_ON dT_dt addp_flux_p log_grad_r ...
       log_grad_p anom_flux_r neoc_flux_r total_flux_r anom_flux_p ...
       neoc_flux_p total_flux_p

% 2D profile time-integration of the Critical Gradient Model (CGM) &
% the Turbulence Spreading Transport Model (TSTM)


% =======CHOOSE SIMULATION MODE=======
% ====================================
% 1 for STEADY-STATE
% 2 for SOURCE MODULATION
% 3 for COLD PULSE
sim_mode = 1;
% ------------------------------------
% 4 for CGM
% 5 for TSTM
modelsim = 4;
% ------------------------------------
% 1 for outboard RF heating
```

```
outboardRF_ON = 1;
% ------------------------------------
% 1 for additional poloidal diffusion
addpol_ON = 1;
% ====================================
% ====================================


% ======= SPECIFY END TIME =================
% calculate profiles for times input below
end_time = 1.1;%[1E-2 5E-2 3E-1 5E-1 2];
% ============== END =======================


% ======= SPECIFY RESOLUTION ===============
% set spat resolution
r_points = 70;
% choose even number of phi grid points (radial profile symmetry)
p_points = 30;
% set timestep tolerances
options  = odeset('RelTol',1e-4,'AbsTol',1e-4,'Stats','on');
% ============== END =======================


% ======= SPECIFY INITIAL CONDITIONS =======
% flat profile initial conditions (edge value for T)
IC_E = 0.1;
IC_T = 7.853e-1;
% ============== END =======================


% ======= SPECIFY EQUATION PARAMETERS ======
% C         : connects heat flux with growth rate
% chi0      : neo-classical (collisional) diffusion
% lambda    : growth rate scaling
% crit_grad : critical gradient
% D0        : turbulence diffusion
% gam0      : turbulence damping
% beta      : turbulence nonlinear (quadratic) saturation
C  = 0.5; chi0 = 0.6; lambda = 1.7; crit_grad = 1.8;
D0 =  35; gam0 = 0.5; beta   =   1;
% ============== END =======================


% ======= RUN SETUP SCRIPT =================
initialise;
% ============== END =======================


% ======= SAFETY FACTOR PROFILE ============
q32 = (1 + r + 6*r.^4).^(3/2);
% ============== END =======================


% ======= COLLECT ALL INPUT IN STRUCTURE====
```

```
 input = struct('r',r,'invdr',invdr,'inv2dr',inv2dr,     ...
               'p',p,'invdp',invdp,'inv2dp',inv2dp,     ...
               'source',source,'parameters',parameters, ...
               'q32',q32,'edgeT',IC_T,'CGM_E',IC_E,     ...
               'modelsim',modelsim,'sim_mode',sim_mode, ...
               'coldpulse',bd_cond,'outboardRF_ON',     ...
               outboardRF_ON);
               %'cp_edgedata',bd_cond,'edgeT',DirichletT, ...
% ============== END =======================

tic
[t,out] = ode23tb(@(t,ET) models2D(t,ET,input),[0 end_time],ICs,options);
toc

if modelsim == 4
    rho = r; T = reshape(out(end,:),length(r),length(p));
else
    rho = r; E = out(:,1:length(r)); T = out(:,length(r)+1:2*length(r));
end

T2D = T;

% % extract cold pulse time traces
% if sim_mode == 3
%     Tcp = zeros(length(t),length(r),length(p));
%     for k = 1:length(t)
%         Tcp(k,:,:) = reshape(out(k,:),length(r),length(p));
%     end
% end


clearvars xcoord ycoord
for k = 1:length(p)
    xcoord(:,k) = r.*cos(p(k));
    ycoord(:,k) = r.*sin(p(k));
end
clearvars Tcoord
Tcoord = T;
% for plotting purposes show also 2*pi ghost point
xcoord(:,length(p)+1) = xcoord(:,1);
ycoord(:,length(p)+1) = ycoord(:,1);
Tcoord(:,length(p)+1) = Tcoord(:,1);
```

D.2 Initialisation Script (initialise.m)

```
% ======= CONSTRUCT RADIAL GRID ============
% step size
dr = 0.87/r_points; invdr = 1./dr; inv2dr = 0.5*invdr;
% rho grid
r = (dr/2:dr:r_points*dr)'; % ghost points included
% ============== END =======================
```

```
% ======= CONSTRUCT POLOIDAL GRID ==========
% step size
dp = 2*pi/p_points; invdp = 1./dp; inv2dp = 0.5*invdp;
% rho grid
p = (0:dp:(p_points−1)*dp)';  %
% ============== END =======================


% ======= SOURCE SETUP =====================
source = struct('NBI_HEATING',[],'RF_HEATING',[],      ...
                'amp',[],'amplitude',[],'boost',[],    ...
                'ICRHmodulation_t',[]);
source.NBI_HEATING = 2/3*(50*(1−r).^2+5);
rhoICRH = 0.32;       %                  <−− ICRH local heating at rhoICRH
source.RF_HEATING  = 70*2/3*exp(−(r−rhoICRH).^2*17^2);
% set asymmetrical heating param. and heat modulation amplitude & frequency
source.amp = 0.75; source.amplitude = 0.55; source.mod_freq = 15;
% start modulation of ICRH source at time:
source.ICRHmodulation_t = 0.7;
% RF part of source radial profile
RF = source.RF_HEATING;
% outboard concentrated RF heating
% ADD dphi/2 IN exp(...) ARGUMENT SO ADDITIONAL POL FLATTENS OUTBOARD PEAK
gaussian = @(x) exp(−4*(x+dp/2−pi).^2);
outboard_RF = trapz([p; 2*pi],gaussian([p; 2*pi]));
relation_RF = zeros(size(r));
for k = 1:length(r), relation_RF(k) = 2*pi*RF(k)/outboard_RF; end
source.relation_RF = relation_RF;
source.gaussian = gaussian;
% ============== END =======================


% % ======= LOAD DATA ========================
% load heat modulation experimental data
[vartitle, moddata] = hdrload('./data/55809_exp_data.txt');
% plasma edge cold pulse Dirichlet data
load ./data/bd_cond.txt
% load experimental data−>used later in spatial q, chi0, etc. investigation
% [vars, expdat] = hdrload('./data/dati_chippy_new.txt');
% % load heat modulation experimental & CGM data
% [vartitle, moddata] = hdrload('./data/55809_exp_data.txt');
% [oldtitle, olddata] = hdrload('./data/55809_CGM.txt');
% % ============== END =======================


% ======= EQUATION PARAMETER ARRAY =========
parameters = [C chi0 lambda crit_grad D0 gam0 beta];
% ============== END =======================


% ======= SETUP INITIAL CONDITIONS =========
if     modelsim == 4 % CGM
    ICs              = zeros(length(r),length(p));
    for k = 1:length(p)
```

```
        ICs(:,k) = IC_T*exp(crit_grad*(r(end)-r));
    end
elseif modelsim == 5 % TSTM
    ICs              = zeros(2*length(r),1);
    ICs(1:end/2)     = IC_E;
    ICs(end/2+1:end) = IC_T*exp(crit_grad*(r(end)-r));
else
    disp('wrong modelsim input: choose ''4'' or ''5''');
    break
end
% ============== END =======================
```

D.3 CGM RHS Function (models2D.m)

```
function [dET_dt] = models2D(t,ET,input)

global S gam_r addpol_ON T chi_r qT32 dT_dt addp_flux_p log_grad_r ...
       log_grad_p anom_flux_r neoc_flux_r total_flux_r anom_flux_p ...
       neoc_flux_p total_flux_p

persistent stepcount plotvar

if isempty(stepcount), stepcount = 0; plotvar = 0; end

stepcount = stepcount+1;
if mod(stepcount,100) == 0
    fprintf('timestep #%u, timestep value = %f\r',stepcount,t)
end

r = input.r;
inv2dr = input.inv2dr;
p = input.p;
invdp = input.invdp;
inv2dp = input.inv2dp;


C         = input.parameters(1);
chi0      = input.parameters(2);
lambda    = input.parameters(3);
crit_grad = input.parameters(4);
D0        = input.parameters(5);
gam0      = input.parameters(6);
beta      = input.parameters(7);

q32 = input.q32;

Nr = length(r)+2; % radial length,  ghost points included
Np = length(p)+2; % poloidal length ghost points included
T = zeros(Nr,Np);

modelsim = input.modelsim;
```

```
sim_mode = input.sim_mode;

no_gr = 2:Nr-1; % index without ghost points
no_gp = 2:Np-1; % index without ghost points

T(no_gr,no_gp) = reshape(ET,Nr-2,Np-2);

% r = 0 ghost points set equal r = 0 physical point dphi = pi away
no_gp_r0_interval = no_gp;
no_gp_r0_interval = circshift(no_gp_r0_interval,(length(no_gp_r0_interval))/2);
T(1,no_gp)  = T(2,no_gp_r0_interval);
%T(1,no_gp) = T(2,no_gp);
T(Nr,no_gp) = 2*input.edgeT-T(Nr-1,no_gp); % inhomogeneous Dirichlet at r = 0.87
T(no_gr,Np) = T(no_gr,2);                  % periodic BC at phi = 0   & 2*pi
T(no_gr,1)  = T(no_gr,Np-1);               % periodic BC at phi = -dp & 2*pi-dp

% COLD PULSE IN 2D POLOIDAL PLANE MODELING
if sim_mode == 3 % use cold pulse data
    bd_cond = input.coldpulse;
    Tss = bd_cond(1,2);
    T_edgeCP  = bd_cond(abs(bd_cond(:,1)-t)==min(abs(bd_cond(:,1)-t)),2);
    cp_gp = length(p)/2:(length(p)/2+2); % COLD PULSE POLOIDAL GRID POINTS
    Nphitotal = length(p); Ncp = length(cp_gp);
    TnoCP = Ncp*(T_edgeCP(1)-Tss)/(Ncp-Nphitotal) + Tss;
    % COLD PULSE EDGE POINTS
%   T(Nr,no_gp) = 2*T_edgeCP(1)-T(Nr-1,no_gp);
    T(Nr,cp_gp) = 2*T_edgeCP(1)-T(Nr-1,cp_gp);
    % ALL BUT COLD PULSE EDGE POINTS
    T(Nr,1:cp_gp(1)-1)     = 2*TnoCP-T(Nr-1,1:cp_gp(1)-1);
    T(Nr,cp_gp(end)+1:end) = 2*TnoCP-T(Nr-1,cp_gp(end)+1:end);
end

qT32 = T(no_gr,no_gp).*sqrt(T(no_gr,no_gp));
% multiply with radially dependent safety factor
q32mat = repmat(q32,1,Np-2);
qT32 = q32mat.*qT32;
chi_collisional = chi0*qT32;


log_grad_r = zeros(Nr,Np); % need ghost points
log_grad_p = zeros(Nr,Np); % need ghost points

% radial part of logarithmic temperature gradient
log_grad_r(no_gr,no_gp) = inv2dr*log(T(3:Nr,no_gp)./T(1:Nr-2,no_gp));

% poloidal part of logarithmic temperature gradient
invrmat = repmat(1./r,1,Np-2);
log_grad_p(no_gr,no_gp) = invrmat*inv2dp.*log(T(no_gr,3:Np)./T(no_gr,1:Np-2));

%gamma_r          = lambda*max(-log_grad(no_g)-crit_grad,-0.05); % upgradient
gamma_r          = lambda*max(-log_grad_r(no_gr,no_gp)-crit_grad,0.0); % Heaviside
gam_r            = sqrt(gamma_r);
posgrowth_r      = logical(imag(sqrt(gamma_r)));
gam_r(posgrowth_r) = -sqrt(-gamma_r(posgrowth_r));
```

```
chi_r = zeros(Nr,Np); % need ghost points

% radial quantities
chi_r(no_gr,no_gp) = C*gam_r.*qT32;
anom_flux_r = chi_r(no_gr,no_gp).*T(no_gr,no_gp).*log_grad_r(no_gr,no_gp);
neoc_flux_r = chi_collisional.*T(no_gr,no_gp).*log_grad_r(no_gr,no_gp);
total_flux_r = anom_flux_r + neoc_flux_r;

% poloidal quantities
anom_flux_p = chi_r(no_gr,no_gp).*T(no_gr,no_gp).*log_grad_p(no_gr,no_gp);
neoc_flux_p = chi_collisional.*T(no_gr,no_gp).*log_grad_p(no_gr,no_gp);
% ADDITIONAL POLOIDAL FLUX
addp_flux_p = 1000*abs(log_grad_p(no_gr,no_gp)).*qT32.*T(no_gr,no_gp) ...
              .*log_grad_p(no_gr,no_gp);
total_flux_p = anom_flux_p + neoc_flux_p;

chi_r(no_gr,no_gp) = chi_r(no_gr,no_gp) + chi_collisional; % total radial chi

% HERE THE ADDITIONAL POLOIDAL DIFFUSSION IS INCLUDED
chi_p(no_gr,no_gp) = chi_r(no_gr,no_gp) + 1000*abs(log_grad_p(no_gr,no_gp)).*qT32;

% ===== DEFINE GHOST POINTS NEEDED FOR EVALUATION OF dT_dt RHS =====

% r = 0 ghost points set equal r = 0 physical point dphi = pi away
chi_r(1,no_gp)  = chi_r(2,no_gp_r0_interval);
% homogeneous Neumann at boundaries
chi_r(Nr,no_gp) = chi_r(Nr-1,no_gp);

% poloidal chi_p is poloidally periodic
chi_p(no_gr,Np) = chi_p(no_gr,2);
chi_p(no_gr,1)  = chi_p(no_gr,Np-1);


% r = 0 ghost points set equal r = 0 physical point dphi = pi away
log_grad_r(1,no_gp)  = log_grad_r(2,no_gp_r0_interval);
% slope continues at r = 0.87
log_grad_r(Nr,no_gp) = 2*log_grad_r(Nr-1,no_gp) - log_grad_r(Nr-2,no_gp);

% poloidal log_grad is poloidally periodic
log_grad_p(no_gr,Np) = log_grad_p(no_gr,2);
log_grad_p(no_gr,1)  = log_grad_p(no_gr,Np-1);

% =====                          END                           =====


FH = 2/3*0.372;


% ================== CRITICAL GRADIENT MODEL RHS ===================

dT_dt = FH*(invrmat.*T(no_gr,no_gp).*chi_r(no_gr,no_gp).*log_grad_r(no_gr,no_gp) ...
            + inv2dr*(T(3:Nr,no_gp)-T(1:Nr-2,no_gp)).*chi_r(no_gr,no_gp) ...
            .*log_grad_r(no_gr,no_gp) ...
            + T(no_gr,no_gp).*inv2dr.*(chi_r(3:Nr,no_gp)-chi_r(1:Nr-2,no_gp)) ...
            .*log_grad_r(no_gr,no_gp) ...
```

```
            +T(no_gr,no_gp).*chi_r(no_gr,no_gp).*inv2dr ...
            .*(log_grad_r(3:Nr,no_gp)-log_grad_r(1:Nr-2,no_gp)) ...
            + invrmat.*(inv2dp*(T(no_gr,3:Np)-T(no_gr,1:Np-2)).*chi_r(no_gr,no_gp) ...
            .*log_grad_p(no_gr,no_gp) ...
            + T(no_gr,no_gp).*inv2dp.*(chi_r(no_gr,3:Np)-chi_r(no_gr,1:Np-2)) ...
            .*log_grad_p(no_gr,no_gp) ...
            +T(no_gr,no_gp).*chi_r(no_gr,no_gp).*inv2dp ...
            .*(log_grad_p(no_gr,3:Np)-log_grad_p(no_gr,1:Np-2))));

if addpol_ON

dT_dt = FH*(invrmat.*T(no_gr,no_gp).*chi_r(no_gr,no_gp).*log_grad_r(no_gr,no_gp) ...
            + inv2dr*(T(3:Nr,no_gp)-T(1:Nr-2,no_gp)).*chi_r(no_gr,no_gp) ...
            .*log_grad_r(no_gr,no_gp) ...
            + T(no_gr,no_gp).*inv2dr.*(chi_r(3:Nr,no_gp)-chi_r(1:Nr-2,no_gp)) ...
            .*log_grad_r(no_gr,no_gp) ...
            +T(no_gr,no_gp).*chi_r(no_gr,no_gp).*inv2dr ...
            .*(log_grad_r(3:Nr,no_gp)-log_grad_r(1:Nr-2,no_gp)) ...
            + invrmat.*(inv2dp*(T(no_gr,3:Np)-T(no_gr,1:Np-2)).*chi_p(no_gr,no_gp) ...
            .*log_grad_p(no_gr,no_gp) ...
            + T(no_gr,no_gp).*inv2dp.*(chi_p(no_gr,3:Np)-chi_p(no_gr,1:Np-2)) ...
            .*log_grad_p(no_gr,no_gp) ...
            +T(no_gr,no_gp).*chi_p(no_gr,no_gp).*inv2dp ...
            .*(log_grad_p(no_gr,3:Np)-log_grad_p(no_gr,1:Np-2))));

end

% load source profile
NBI   = input.source.NBI_HEATING;
RF    = input.source.RF_HEATING;
amp   = input.source.amp;
mod_t = input.source.ICRHmodulation_t;
mod_f = input.source.mod_freq;

if sim_mode == 2 && t>mod_t
    amplitude     = input.source.amplitude;
    P_mod_off     = amp-amplitude; % switch off asymmetrical heating
    P_mod_on      = 2*amplitude;   % switch on  asymmetrical heating
    pulse_factor  = t*mod_f;
    fpulse_factor = floor(pulse_factor);
    onORoff       = pulse_factor-fpulse_factor;
    amp           = P_mod_off + heaviside(onORoff*2-1)*P_mod_on;
end

if input.outboardRF_ON % poloidally asymmetric RF heating

    outboard_RF = input.source.gaussian(p);          % at phi = pi
    %findphi0 = find(outboard_RF==max(outboard_RF)); % at phi =  0
    %outboard_RF = repmat(circshift(outboard_RF,-findphi0+1)',Nr-2,1);
    outboard_RF = repmat(outboard_RF',Nr-2,1);
    relation_RF = repmat(input.source.relation_RF,1,Np-2);
    RF = relation_RF.*outboard_RF;
    NBI = repmat(NBI,1,Np-2);
    S = NBI+amp*RF;
    %S = relation_S.*outboard_S;
```

```
else                   % poloidally  symmetric RF heating

    S = NBI+amp*RF;
    S = repmat(S,1,Np-2);

end

% add source
span = 1:length(r)-1; % no source on edge (no material to heat)
                      % ((and undermines T edge BC definition))

dT_dt(span,:) = dT_dt(span,:)+S(span,:);
dET_dt = dT_dt(:);
```

D.4 Heat Modulation Analysis

```
tic
tdata_after = 0.8;%seconds
index = find(t>tdata_after);
% ----------------HARDCODED-FREQUENCY------------------
     pulse_length = 1/input.source.mod_freq;
% -----------------------------------------------------
rho = r;
% look at phi = 0 [p(1) = 0]
phi_index = length(p)/2+2; % = pi, highest concentration of outboard heating
T = zeros(length(t),length(r));
for k = 1:length(t)
    temp = reshape(out(k,:),length(r),length(p));
    T(k,:) = temp(:,phi_index);
end
U = T;
% spatial modulation source index
RFindex = find(abs(rho-rhoICRH)==min(abs(rho-rhoICRH)));%+length(rho);
% index span one pulselength back from t(end)
pindex1 = find(t>end_time-pulse_length);
% index for minimum in pindex1 at (close to) source
latebound = find(U(pindex1,RFindex)==min(U(pindex1,RFindex)));
% index span two pulse_lengths back
pindex2 = find(t>end_time-2*pulse_length);
pindex3 = linspace(pindex2(1),pindex1(1),pindex1(1)-pindex2(1)+1);
earlybound = find(U(pindex3,RFindex)==min(U(pindex3,RFindex)));
pindex4 = linspace(pindex3(earlybound),pindex1(latebound), ...
                   pindex1(latebound)-pindex3(earlybound)+1)';
% ------------ MAKE DATA PERIODIC ------------
% index span one pulselength forward from tdata_after
sindex1 = find(t(index)<tdata_after+pulse_length);
sindex1 = sindex1+index(1)-1;
% index for minimum in pindex1 at (close to) source
startbound = find(U(sindex1,RFindex)==min(U(sindex1,RFindex)));
% index span two pulse_lengths back
```

```
sindex2 = find(t<tdata_after+2*pulse_length);
sindex3 = linspace(sindex1(end),sindex2(end),sindex2(end) ...
                   -sindex1(end)+1);
endbound   = find(U(sindex3,RFindex)==min(U(sindex3,RFindex)));
sindex4 = linspace(sindex1(startbound),sindex3(endbound) ...
                   ,sindex3(endbound)-sindex1(startbound)+1)';
%figure(3)
%plot(t(sindex4(1):pindex4(end)),U(sindex4(1):pindex4(end),end/2+32))
% ----------------------------------------------------------
periodt = t(sindex4(1):pindex4(length(pindex4)));
periodU = zeros(length(periodt),length(rho));
for spat = 1:length(rho)
    periodU(:,spat) = U(sindex4(1):pindex4(length(pindex4)),spat);
end
% ------ INTERPOLATE DATA AND SHIFT TO t=0 ------
clearvars intpolt intpolU
intpolt = 0:(periodt(end)-periodt(1))/2^14:periodt(end)-periodt(1);
for spat = 1:length(rho)
    intpolU(:,spat) = spline(periodt(:)-periodt(1) ...
                             ,periodU(:,spat),intpolt);
end
% ---------------FOURIER TRANSFORM---------------
% define end time M
M = intpolt(end); % (shifted by 0.7)
% define N for the N-point discrete Fourier Transformation
N = 2^14; %length(intpolt)-1;

% --------------------- TRANSFORM FUNCTIONS ---------------------
DFT = @(k,s) M/(length(index)*sqrt(2*pi))*sum(U(index,s) ...
                .*exp(-2*pi*1i*t(index)*k/M));
trancos = @(k,s) sum(U(index,s).*cos(2*pi*t(index)*k/M));
transin = @(k,s) sum(U(index,s).*sin(2*pi*t(index)*k/M));

% 1000 to get eV rather than keV, the rest of factor in front of sum is
% cos^2 normalisation
intpolDFT = @(k,s) 1000/(2*sum(cos(2*pi*intpolt(:)*k/M).^2)) ...
                   *sum(intpolU(:,s).*exp(-2*pi*1i*intpolt(:)*k/M));
trancosintpol = @(k,s) 1000/(2*sum(cos(2*pi*intpolt(:)*k/M).^2)) ...
                       *sum(intpolU(:,s).*cos(2*pi*intpolt(:)*k/M));
transinintpol = @(k,s) 1000/(2*sum(cos(2*pi*intpolt(:)*k/M).^2)) ...
                       *sum(intpolU(:,s).*sin(2*pi*intpolt(:)*k/M));
NORM = @(k) sum(cos(2*pi*intpolt(:)*k/M).^2);

DFTsimple = @(k) sum(sin(2*pi./(t(index(end))-t(index(1)))*t(index)) ...
                 .*exp(-2*pi*1i*t(index)*k/M));
DFTsimpleintpol = @(k) sum(sin(2*pi/(intpolt(end)-intpolt(1))*intpolt) ...
                       .*exp(-2*pi*1i*intpolt*k/M));
% ---------------------- END OF TRANSFORMS ----------------------


figure(5)
subplot(2,1,1); plot(t(index),U(index,RFindex))
title('Direct data')
xlabel('Time [s]')
ylabel('Temperature [eV]')
```

```
subplot(2,1,2); plot(intpolt,intpolU(:,RFindex))
title('Interpolated data')
xlabel('Time [s]')
ylabel('Temperature [eV]')

clearvars kk transform normalise transformintpol costrans sintrans ...
          intpolcos intpolsin
% which spatial index to look at
spatind = RFindex; %round(rhoICRH*100); % =32 is near heat modulation source
for ooo = 1:100
    kk(ooo) = ooo/M;
    transform(ooo) = DFT(ooo,spatind);
    transformintpol(ooo) = intpolDFT(ooo,spatind);
    costrans(ooo) = trancos(ooo,spatind);
    sintrans(ooo) = transin(ooo,spatind);
    intpolcos(ooo) = trancosintpol(ooo,spatind);
    intpolsin(ooo) = transinintpol(ooo,spatind);
%    simple(ooo) = DFTsimple()
end

figure(12)
subplot(3,1,1); plot(kk,intpolcos)
title('Cos transform of interpolated data')
xlabel('Frequency [Hz]')
subplot(3,1,2); plot(kk,intpolsin)
title('Sine transform of interpolated data')
xlabel('Frequency [Hz]')
subplot(3,1,3); plot(kk,abs(intpolcos+1i*intpolsin))
title('|cos^2+i\cdotsin| for interpolated data')
xlabel('Hz')
ylabel('Amplitude [eV]')

figure(11)

for harmonic = 1:2

    if harmonic == 1
    % look at data in space for 15Hz (fundamental mode)
        evalk = input.source.mod_freq*M;
    else
    % third harmonic
        evalk = input.source.mod_freq*M*3;
    end

    for s = 1:length(rho)
        DFTintpolrho(s) = intpolDFT(evalk,s);
        DFTrho(s) = DFT(evalk,s);
        cosinespat(s) = trancosintpol(evalk,s);
        sinespat(s) = transinintpol(evalk,s);
    end
    leg{1} = sprintf('Fundamental Mode');
    leg{2} = sprintf('3rd Harmonic');
    col{1} = sprintf('b');
    col{2} = sprintf('r');
```

```
    cgmamp(harmonic,:)   = abs(DFTintpolrho);
    cgmphase(harmonic,:) = 100+360/(2*pi)*asin(imag(DFTintpolrho) ...
                           ./abs(DFTintpolrho));

    hold on
    subplot(1,2,1); plot(rho,cgmamp(harmonic,:),col{harmonic},'LineWidth',2)
    title('Amplitude')
    xlabel('\rho [m]')
    ylabel('[eV]')
    axis([0 1 0 65])
    legend(leg{1:harmonic})
    hold on
    subplot(1,2,2); plot(rho(1:end-1),cgmphase(harmonic,1:end-1) ...
                         ,col{harmonic},'LineWidth',2)
    title('Phase')
    xlabel('\rho [m]')
    ylabel('[eV]')
    axis([0 1 50 130])
    legend(leg{1:harmonic},'Location','Northwest')

end
hold on
subplot(1,2,1); plot(moddata(:,1),moddata(:,2),'b.')
hold on
subplot(1,2,1); plot(moddata(:,1),moddata(:,3),'r.')
hold on
subplot(1,2,2); plot(moddata(:,1),moddata(:,4),'b.')
hold on
subplot(1,2,2); plot(moddata(:,1),moddata(:,5),'r.')

toc

lookat = find(t<1.8799);
Tphamp = reshape(out(lookat(end),:),length(r),length(p));
premodindex = find(t<source.ICRHmodulation_t);
Tpremod = reshape(out(premodindex(end),:),length(r),length(p));
figure(97)
subplot(1,2,1); surfc(xcoord,ycoord,Tphamp-Tpremod)
subplot(1,2,2); surf(xcoord,ycoord,Tphamp-Tpremod);
%set(s, 'facecolor', 'none')

%surfc(xcoord,ycoord,Tphamp-Tpremod,'EdgeColor','none', ...
       'LineStyle','none','FaceLighting','phong');

phampWRITEDATA
```

D.5 Cold Pulse Analysis

```
    index = find(t>0.98);
rho = r;
    flue = [round(length(rho)*0.11/rho(end)) round(length(rho)*0.30/rho(end)) ...
```

```
            round(length(rho)*0.60/rho(end)) round(length(rho)*0.86/rho(end))];

% look at phi = 0 [p(1) = 0]
phi_index = length(p)/2+1; % phi = pi, peak asymmetric source concentration
T = zeros(length(t),length(r));
for k = 1:length(t)
    temp = reshape(out(k,:),length(r),length(p));
    T(k,:) = temp(:,phi_index);
end
    pulsestart = find(t>1.012); pulseend = find(t>1.09);
    for dummy = 1:length(rho)
        substravrgALL(dummy) = T(pulsestart(1)-1,dummy) - 0.03; %...
                       %    0.5*(T(index(1),dummy)+T(index(end),dummy)));
        pulselocALL         = find(abs(T(index,dummy)-substravrgALL(dummy))...
                              ==min(abs(T(index,dummy)- ...
                                    substravrgALL(dummy))));
        pulsetlocALL(dummy) = t(index(pulselocALL));
    end
    delayMS = 1000*(pulsetlocALL-min(pulsetlocALL))';
    figure(3)
    subplot(1,2,1); plot(t(index),T(index,flue))
    xlabel('Time [s]')
    ylabel('Plasma Temperature [keV]')
    title('Cold Pulse Footprint')
    subplot(1,2,2); plot(rho,delayMS)
    xlabel('\rho [m]')
    ylabel('Pulse Temporal Location [ms]')
    title('Pulse Propagation')
```

Bibliography

[1] Ailin Parsa and Magnus D. Hansen. Bachelorprojekt: Turbulens og Transport i Magnetiserede Plasma. Technical University of Denmark, 2009.

[2] Naklé H. Asmar. *Partial Differential Equations: with Fourier Series and Boundary Value Problems.* PEARSON, Prentice Hall, ©2005.

[3] R. E. Bank, JR. W. M. Coughran, W. Fichtner, E. H. Grosse, D. J. Rose, and R. K. Smith. Transient simulation of silicon devices and circuits. *IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN*, CAD-4(4), OCTOBER 1985.

[4] C. M. Braams and P. E. Stott. *Nuclear Fusion: Half a Century of Magnetic Confinement Fusion Research.* Intitute of Physics Publishing, ©2002.

[5] D. del Castillo Negrete, P. Mantica, V. Naulin, J. J. Rasmussen, and JET EFDA contributors. Fractional diffusion models of non-local perturbative transport: numerical results and application to JET experiments*. *Nucl. Fusion*, 48(075009):13pp, Jun 2008.

[6] Sohan Dharmaraja. An analysis of the tr-bdf2 integration scheme. Master's thesis, School of Engineering, MASSACHUSETTS INSTITUTE OF TECHNOLOGY, 2007.

[7] Thomas J. Dolan. *Fusion Research: Principles, Experiments and Technology.* Pergamon Press, ©1982 (Thomas J. Dolan ©2000).

[8] Jeffrey P. Freidberg. *Plasma Physics and Fusion Energy.* Cambridge University Press; 1st edition, ©2008.

[9] X. Garbet, P. Mantica, F. Ryter, G. Cordey, F. Imbeaux, C. Sozzi, A. Manini, E. Asp, V. Parail, R. Wolf, and the JET EFDA Contributors. Profile stiffness and global confinement. *Plasma Phys. Control. Fusion*, 46:23pp, Jul 2004.

[10] X. Garbet, Y. Sarazin, F. Imbeaux, P. Ghendrih, and C. Bourdelle. Front propagation and critical gradient transport models. *PHYSICS OF PLASMA*, 14:122305, Dec 2007.

[11] K. W. Gentle, W. L. Rowan, R. V. Bravenec, G. Cima, T. P. Crowley, H. Gasquet, G. A. Hallock, J. Heard, A. Ouroua, P. E. Phillips, D. W. Ross, Schoch P, M, and C. Watts. Strong nonlocal effects in a tokamak perturbative transport experiment. *PHYSICAL REVIEW LETTERS*, 74(18), May 1995.

[12] Jan S. Hesthaven and Tim Warburton. *Nodal Discontinuous Galerkin Methods: Algorithms, Analysis, and Applications.* Texts in Applied Mathematics. Springer Publishing Company, Incorporated, ©2007.

[13] G. T. Hoang, C. Bourdelle, X. Garbet, G. Giruzzi, T. Aniel, M. Ottaviani, W. Horton, P. Zhu, and R. V. Budny. Experimental determination of critical threshold in electron transport on tore supra. *Phys. Rev. Lett.*, 87(12):125001, Sep 2001.

[14] http://www.iter.org/.

[15] F. Imbeaux, F. Ryter, and X. Garbet. Modelling of ECH modulation experiments in ASDEX Upgrade with an empirical critical temperature gradient length transport model. *Plasma Phys. Control. Fusion*, 43, 2001.

[16] A. Jacchia, F. De Luca, F. Ryter, A. Bruschi, F. Leuterer, R. Neu, G. Pereverzev, W. Suttrop, D. Wagner, and ASDEX Upgrade Team. Nonlinear perturbative electron heat transport study in the ASDEX Upgrade tokamak. *Nucl. Fusion*, 45:40–47, 2005.

[17] J. E. Kinsey, R. E. Waltz, and H. E. St. John. Theoretical transport modeling of Ohmic cold pulse experiments. *PHYSICS OF PLASMAS*, 5(11), Nov 1998.

[18] Randall J. Leveque. *Finite Volume Methods for Hyperbolic Problems.* Cambridge University Press, ©2002.

[19] Randall J. Leveque. *Finite Difference Methods for Ordinary and Partial Differential Equations: Steady-State and Time-Dependent Problems.* Society for Industrial and Applied Mathematics (SIAM), ©2007.

[20] Z. Lin, S. Ethier, T. S. Hahm, and W. M. Tang. Size scaling of turbulent transport in magnetically confined plasmas. *Phys. Rev. Lett.*, 88(19):195004, Apr 2002.

[21] P. Mantica, I. Coffey, R. Dux, X. Garbet, L. Garzotti, G. Gorini, F. Imbeaux, E. Joffrin, J. Kinsey, M. Mantsinen, R. Mooney, V. Parail, Y. Sarazin, C. Sozzi, W. Suttrop, G. Tardini, D. Van Eester, and JET EFDA contributors. Transient heat transport studies in JET conventional and advanced tokamak plasmas. *EX/P1-04*, 2002.

[22] P. Mantica, G. Corrigan, X. Garbet, F. Imbeaux, J. Lonnroth, V. Parail, T. Tala, A. Taroni, M. Valisa, and H. Weisen. Chapter 10: Core transport studies in jet. *FUSION SCIENCE AND TECHNOLOGY*, 53, May 2008.

[23] P. Mantica, G. Gorini, F. Imbeaux, J. Kinsey, Y. Sarazin, R. Budny, I. Coffey, R. Dux, X. Garbet, L. Garzotti, C. Ingesson, M. Kissick, V. Parail, C. Sozzi, A. Walden, and contributors to the EFDA-JET Workprogramme. Perturbative transport experiments in JET low or reverse magnetic shear plasmas*. *Plasma Phys. Control. Fusion*, 44:2185–2215, Sep 2002.

[24] P. Mantica and F. Ryter. Perturbative studies of turbulent transport in fusion plasmas. *C. R. Physique*, 7:643–649, Aug 2006.

[25] Paola Mantica. Private communication.

[26] MathWorks. MATLAB documentation.

[27] Garry McCracken and Peter Stott. *Fusion: The Energy of the Universe.* Elsevier Inc., ©2005.

[28] V. Naulin, J. J. Rasmussen, P. Mantica, D. del Castillo-Negrete, and JET-EFDA contributors. Fast heat pulse propagation by turbulence spreading. *J. Plasma Fusion Res. SERIES*, Vol. 8, 2009.

[29] Volker Naulin. Private communication.

[30] Arthur G. Peeters, Clemente Angioni, and Giovanni Tardini. Transport modelling. *Comptes Rendus Physique*, 7(6):592 – 605, 2006. Turbulent transport in fusion magnetised plasmas.

[31] Robert D. Richtmyer and K. W. Morton. *Difference Methods for Initial-Value Problems.* Interscience Tracts in Pure and Applied Mathematics. John Wiley & Sons, Inc., ©1967.

[32] F. Ryter, F. Imbeaux, F. Leuterer, H.-U. Fahrbach, W. Suttrop, and ASDEX Upgrade Team. Experimental characterization of the electron heat transport in low-density ASDEX Upgrade plasmas. *Phys. Rev. Lett.*, 86(24):5498–5501, Jun 2001.

[33] F. Ryter, R. Neu, R. Dux, H.-U. Fahrbach, F. Leuterer, G. Pereverzev, J. Schweinzer, J. Stober, W. Suttrop, and ASDEX Upgrade Team. Propagation of cold pulses and heat pulses in ASDEX Upgrade. *Nuclear Fusion*, 40(11), 2000.

[34] Murray R. Spiegel. *Schaum's Outlines: Fourier Analysis with Applications to Boundary Value Problems.* McGraw-Hill Companies, inc., ©1974.

[35] Steven H. Strogatz. *Nonlinear Dynamics And Chaos: With Applications To Physics, Biology, Chemistry, And Engineering.* Westview Press; 1st edition, ©2001.

[36] Eleuterio F. Toro. MUSTA: A MULTI-STAGE NUMERICAL FLUX. *http://www.ing.unitn.it/toro.*

[37] Eleuterio F. Toro. *Riemann Solvers and Numerical Methods for Fluid Dynamics.* Berlin: Springer Verlag, ©1999.

[38] F. Wagner, A. Becoulet, R. Rudny, V. Erckmann, D. Farina, G. Giruzzi, Y. Kamada, A. Kaye, F. Koechl, K. Lackner, N. Marushchenko, M. Murakami, T. Oikawa, V. Parail, J. M. Park, G. Ramponi, O. Sauter, D. Stork, P. R. Thomas, Q. M. Tran, D. Ward, H. Zohm, and C. Zucca. On the heating mix of ITER. *Plasma Phys. Control. Fusion*, 52:124044, Nov 2010.