Summer 8-2020

Negative-Triangularity Configuration on EAST: Analysis of engineering limitations on superconducting, D-shaped, target-diverted plasmas

David A. Weldon

Negative-Triangularity Configuration on EAST:

Analysis of engineering limitations on superconducting, D-shaped, target-diverted plasmas

A thesis

Submitted to the Factulty

of

Rose-Hulman Institute of Technology

by

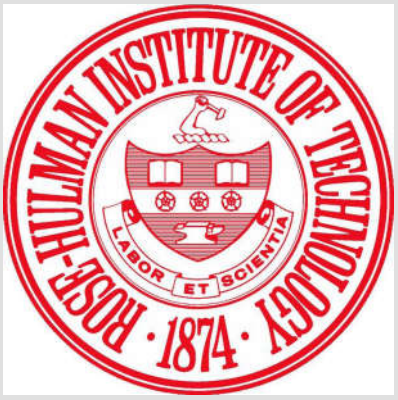
David A. Weldon

In Partial Fulfillment of the Requirements for the Degree

of

Masters of Science in Electrical Engineering

August 2020



ROSE-HULMAN INSTITUTE OF TECHNOLOGY

Final Examination Report

David Weldon — Name

Electrical Engineering — Graduate Major

Thesis Title: Negative-Triangularity Configuration on EAST: Analysis of Engineering Limitations on Superconducting, D-Shaped, Target-Diverted Plasma

DATE OF EXAM: August 21, 2020

EXAMINATION COMMITTEE:

	Thesis Advisory Committee	Department
Thesis Advisor:	Clifford Grigg	ECE
	Robert Throne	ECE
	Edward Wheeler	ECE
	Scott Kirkpatrick	PHOE

PASSED X **FAILED** _____

ABSTRACT

Weldon, David A.

M.S.E.E

Rose-Hulman Institute of Technology

August 2020

Negative-Triangularity Configuration on EAST: Analysis of engineering limitations on superconducting, D-shaped, target-diverted plasma

Thesis Advisor: Dr. Cliff Grigg

Thermonuclear fusion is so named because of the high temperature that the majority of the fuel must maintain such that nuclei can overcome the electrostatic force, fuse, and produce energy. However, the ions and electrons (plasma) are so hot that any material used to confine them would be destroyed. To achieve confinement while maintaining the 50,000,000 K temperature needed for self-sustaining fusion, magnetic confinement is needed. As of 2019, the tokamak is the leading candidate for a practical fusion reactor.

In recent years, tokamak research has repeatedly shown that the edge magneto-hydrodynamic stability is critical for handling the power to the walls and the divertor plates which is now and will most likely continue to be a limiting factor in the International Thermonuclear Experimental Reactor (ITER) and the DEMOnstration Power Station (DEMO). Recent experiments at Tokamak à Configuration Variable (TCV) and DIII-D have shown that a Negative-Triangularity Configuration (NTC) has a larger power handling area on the Low-Field-Side (LFS) divertor target plate and improved edge stability. However, there have been relatively few NTC experiments performed so far and none of them have been performed on a superconducting tokamak with shaping capabilities similar to ITER. To expand upon the previous experiments on TCV and DIII-D this thesis addresses an initial test of the NTC capability of the Experimental Advanced Superconducting Tokamak (EAST) which has achieved a > 6 s ohmic discharge Upper Singular Null (USN) target-diverted plasma with a lower triangularity of $\delta_L \leq -0.09$.

Dedication

In loving memory of my mother.

None of this work would be possible without the continued encouragement and support of my parents, Richard and Elizabeth Weldon.

Declaration

I declare this thesis has been composed solely by myself and that it has not been submitted, in whole or in part, in any previous application for a degree. Except where states otherwise by reference or acknowledgment, the work presented is entirely my own.

Acknowledgments

First and foremost I'd like to thank my advisor Dr. XIAO Bingjia (ASIPP, EAST) for taking me on as his graduate student and the incredible patience he has given me during the course of my work, Dr. LUO Zhengping (ASIPP, EAST) for his tireless efforts to explain and teach the finer details of EAST operation to a hard-headed graduate student. Thanks to Dr. Anders Welander (GA, DIII-D) for his email and text collaboration and the countless hours he spent in painstakingly detailed explanations and instructions. And a thanks to the ASIPP computer controls division team and the General Atomics DIII-D team. This work was supported by the National MCF Energy R&D Program of China (Grant No. 2018YFE0302100), the National Natural Science Foundation of China (Grant No. 11875291), and the tuition grant and graduate assistanceship of Rose-Hulman Institute of Technology.

Just as importantly, I'd like to thank my advisor Dr. Cliff Grigg who took me on as a graduate student despite my lack of direction all those year ago. Not only has he been a constant source of support and encouragement, but he is one of the finest teachers Rose-Hulman has to offer. There are countless other professional and personal friends who have helped a great deal in technical matters as well as in emotional support. Not least of which is Steven Slaven, my middle school science teacher whose dedication to the Science Olympiad program initiated my pursuit and dedication to scientific discovery. My desire for learning and understanding was accelerated thanks to the passionate teachers at Saint Lawrence Seminary High School and Rose-Hulman Institute of Technology. My friends and brothers in $\phi\phi\kappa\alpha$, especially John Sullivan, Brian Edmonson, and Mark Tappendorf. Brent Covele deserves special recognition for his part in starting me up the long walk to fusion

physics and tokamak research, despite how his views have changed over time. XU Feng also deserves to be singled out for all his help and support in tutoring me through my classes in China. Without his help, I would not have passed a single class. And finally, to my Peace Corps family who have added encouragement through their own perseverance and unrelenting efforts in their own lives which in turn encourages me to do the same.

Contents

List of Figures

List of Tables

List of Listings

Chapter 1

Introduction

1.1 The future of world energy consumption

1.1.1 Current energy consumption

The human population and its technologies continue to grow at an astounding rate. The rapid population growth in concert with the ever-increasing standard of living, especially in developing countries, now presents humanity with one of its greatest challenges to date: sustainable energy production. Worldwide electricity consumption has grown from 10.9 petawatt-hours (1 PWh = 10^{15} Wh) in 1990 to more than 23.7 PWh in 2017 [1] and is expected to continue on this trend for the foreseeable future. Traditionally, fossil fuels i.e. coal, oil, and natural gas have been the primary energy source to meet the ever-growing energy demand. However, the combustion of these fuels come at ever-growing costs: air pollution, diminishing fuel reserves, harmful mining practices, and increasingly likely an irrevocable negative impact on the global climate. An approach to mitigating these costs is to replace fossil fuels with renewable energy sources such as wind and solar, and while they do show promise for power generation many questions remain unanswered concerning efficiency, storage, regional availability, and reliability for producing sufficient baseload electricity.

Another alternative is nuclear fission which produces energy by splitting the large nuclei of heavy elements into smaller, lighter elements. In 2017 nuclear fission provided 2.6 PWh of energy

worldwide [1] and is expected to grow. In comparison to conventional fossil fuels, the energy density of a nuclear reaction is staggering: the fission of U-235 releases 8e13 J/kg while a typical chemical reaction only yields 5×10^7 J/kg. The incredible energy density of fission reactors makes them an attractive alternative to fossil fuel power plants. However, fission is not without its negative aspects as well. Three Mile Island (1979), Chernobyl (1986), Tokaimura (1999), and most recently Fukushima Daiichi (2011) have all been major nuclear disasters that released large amounts of radioactive materials despite the advances made in safety technology. On top of the concerns of future disasters, there is the question of how to handle nuclear waste which is highly radioactive with a half-life on the order of 1 million years. The proposals for deep underground storage or developing and using fast-fission reactors for disposal have yet to find favor with all those concerned, leaving the waste to accumulate at its respective nuclear power plant in most cases. Another important consideration is that of nuclear proliferation. The technology used to enrich naturally occurring uranium ($\sim$ 0.7% U-235) to reactor-grade ($\sim$ 5% U-235) is, in principle, the same as that used for enriching uranium to weapons-grade ($\sim$ 85% U-235). To encourage the world to use fission energy as the main replacement for fossil fuels may also encourage an unacceptable risk of nuclear weapons proliferation.

1.1.2 Fusion energy

Since 1942, when a discussion with Enrico Fermi prompted Edward Teller to perform the first investigation into fusing two deuterium nuclei, the dream of nuclear fusion producing clean, cheap energy has driven thousands of man-years of research. When realized, fusion power will supply much of the world's energy demand without the severe downsides of the previously discussed energy sources. In contrast to fission, fusion produces energy by taking light elements with small nuclei and combines them to form a larger, heavier nucleus. The best example of this is the sun, which is essentially an ongoing fusion reaction producing approximately 4×10^{26} watts of continuous power. The particular breed of fusion that the sun utilizes is a simple proton-proton reaction which is not particularly efficient, at least among fusion reactions, even in the most active regions of the sun the

power density is only about 270 W/m^3 [2]. The sun also relies on its enormous mass ($\sim 2 \times 10^{30}$ kg) which creates enough gravity to sufficiently confine the protons to keep the reaction going. This proton-proton reaction and gravitational confinement are obviously not suitable for terrestrial fusion reactors. Fortunately, the proton-proton reaction is not the only reaction available to us on Earth, not only do others exist, but they are more efficient as well. The most promising of which is the deuterium (D, hydrogen with one neutron) and tritium (T, hydrogen with two neutrons) reaction:

$$ {}^{2}_{1}\mathrm{D} + {}^{3}_{1}\mathrm{T} \longrightarrow {}^{4}_{2}\mathrm{He} + {}^{1}_{0}\mathrm{n} + 17.6\,\mathrm{MeV} \tag{1.1} $$

This reaction has an energy density nearly three times that of the U-235 reaction, yielding 3×10^{14} J/kg. To put this into power generation numbers, about 40 kg of deuterium and 60 kg of tritium would supply a 1 GW (thermal) fusion power plant with 1 year of fuel. While deuterium occurs naturally it is only at a concentration of 0.02% of all hydrogen atoms on earth, however that is still a nearly unlimited supply when multiplied by the billions of cubic kilometers of water in the world's oceans. Tritium, on the other hand, does not occur naturally in any usable concentrations and so it must be produced as needed as its half-life is only 12.32 years via beta decays yielding He-3 [3]. The most promising method of producing tritium at the moment is breeding from Li-6 using neutron activation. Li-6 is a naturally occurring isotope of lithium and is just 7.4% of mined Li but given that there are an estimated 230 billion tones of Li in the Earth's crust and oceans, Li-6 is virtually unlimited [4]. A breeding-blanket has been proposed in many fusion reactor designs which would require a layer of Li-6 in the reactor's vacuum vessel that would be activated by the fast neutrons produced in a deuterium-tritium reaction. Therefore, the fuel supply for reactors utilizing deuterium-tritium fusion is practically unlimited. Furthermore, the relative cost of deuterium and tritium is low compared to the amount of energy they can produce, making fusion reactors especially promising for producing baseline power on a large scale.

If these selling points were not enough, the deuterium-tritium reaction does not emit any greenhouse gases, air pollutants, or toxic chemicals. The main "waste" product is helium which can

be easily captured for industrial use. Unlike fission reactions, there is no risk of a meltdown because runaway fusion reactions are impossible, since the reaction abruptly ends once ideal conditions are disrupted. However, the fast neutrons produced from the deuterium-tritium reaction will activate the surrounding materials over time and will then need to be handled as radioactive waste once they have reached the end of their life-cycle; that being said, these activated materials have a half-life of around 50 years at most so that advanced, long-term storage is not necessary. Finally, the technology used in a fusion reactor is entirely unrelated to the technology needed for making fusion weapons. This unfortunately does not mean proliferation is completely negated. Fusion reactors using the deuterium-tritium reaction produce high energy neutron beams that could be used for the production of weapons-grade plutonium-239 from the very abundant uranium-238 [5]. This has yet to be a significant consideration, however, since a scheme for harnessing these neutron beams has not been fully designed, let alone tested or implemented. Furthermore, there are currently simpler and more efficient technologies for plutonium production. Notwithstanding, this is still something to be considered in a future where fusion plays a major role in energy production.

While the pros and cons discussed above may still make fusion sound like a relatively good energy source, it is not without its technical issues: the prohibitive cost of a reactor and extreme difficulty of creating a net-positive reaction. When in a plasma state, both deuterium and tritium have a +1 electric charge, thus a large electrostatic force acts as a barrier to their fusion. By random thermal motion at room temperature, the fusion reaction rate is nearly non-existent and only increases to a usable level when the temperature exceeds 120 million řC. At this high temperature, the deuterium and tritium nuclei have sufficient energy to overcome the Coulomb force to get close enough together for the quantum tunneling effect via the strong nuclear force to bind them together. As seen in Figure 1.1 the temperature needs to reach 10 keV; it is best to use the convention in plasma physics which is to express temperatures in electron-volts. This energy is equivalent to the temperature in řC mentioned above. Surprisingly enough, this is several times hotter than the core of the sun and is achievable in both magnetic and inertial confinement fusion on Earth. This extreme temperature is magnitudes hotter than any substance available can withstand which is why

the aforementioned confinement methods are used to keep the plasma contained. These extreme temperatures reduce all materials to their constituent elements, the nuclei of which are also stripped of their electrons leaving a mix of positively-charged ions and negatively-charged electrons that behave similarly to a quasi-neutral, electrically conductive gas. For the extent of this thesis, this will be the definition of plasma, the fourth state of matter.

1.1.3 The triple product

Confinement of the plasma does not just mean that the particles are contained, it also means that the energy is contained as well. Fusion reactions have been demonstrated in laboratory conditions since Mark Oliphant first accomplished the fusion of hydrogen isotopes in 1932. However, these types of fusion reactions consume far more energy than they produce. Fusion is said to reach *ignition* when the energy from the reaction keeps the fuel energetic enough to continue the fusion reaction, roughly speaking. The conditions necessary for ignition are different for each fusion device so to have a comparison of their conditions, a metric of some sort is needed. John Lawson first wrote about the minimum requirements for plasma electron density, n_e, and the plasma energy confinement time, τ_E, and was later expanded upon to include the plasma temperature, T, together known as the *triple product* though it is still commonly called the Lawson Criterion.

For the most common reaction of deuterium and tritium a brief derivation is given. The thermal energy density (the energy per volume) W in the plasma is

$$W = \frac{3}{2}\left(n_D + n_T + n_e\right) k_B T \tag{1.2}$$

where n_D is the number density of the deuterium ions, n_T is the number density of the tritium ions, n_e is the number density of the electrons, k_B is the Boltzmann constant, and T is the temperature in Kelvin. Note, for the remainder of Section 1.1.3 T will refer to temperatures in Kelvin unless otherwise stated. Assuming there is an equal numbers of neutral deuterium and tritium atoms to

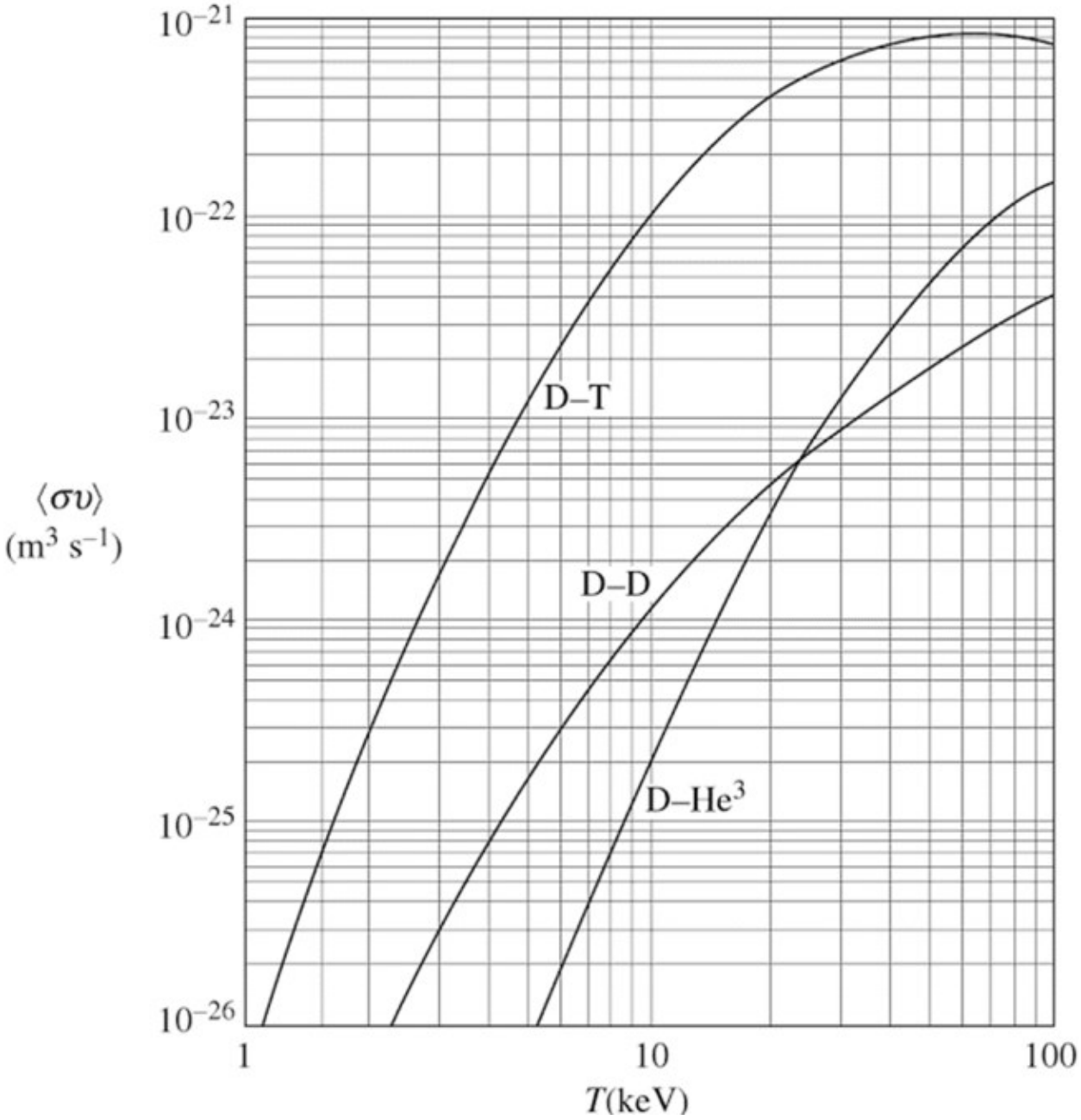


Figure 1.1: Volumetric reactions rates of D-T, D-D, and D-He^3 as a function of plasma temperature, where σ is the ion cross-section and v is the ion velocity, and the fusion reaction rate is given as $\langle \sigma v \rangle$ is averaged over a Maxwellian distribution [6].

start with, then

$$n_D = n_T = \frac{n_e}{2} \tag{1.3}$$

and by substituting $n = n_e$, this simplifies Equation (1.2) to

$$W = 3nk_BT. \tag{1.4}$$

The energy balance of any confinement scheme then becomes

$$\dot{W} = P'_{fus} + P_{in} - P_{loss} \tag{1.5}$$

where P'_{fus} is the fraction of fusion power density that remains in the plasma system, P_{in} is the power inject into the plasma system, and P_{loss} represents all the power losses of the system.

The power loss can be attributed to two main causes: particle loss and brems-strahlung radiation. The particle losses also carry with them a large amount of the thermal energy of the plasma and are a major consideration. The bremsstrahlung radiation occurs when energetic electrons interact with other particles in the plasma and decelerate. In general, this can be given by

$$P_B = 1.69 \times 10^{-38} n^2 Z_i^2 k_B T^{\frac{1}{2}} \tag{1.6}$$

where P_B is the power loss due to bremsstrahlung radiation and Z_i is the ion charge which for all deuterium-tritium reaction is 1.

The nuclear fusion reaction produces power density as a function of the ion density and the plasma temperature itself. If the temperature is taken to be the thermal motion of the ions then a plasma at a higher temperature will have ions with enough energy to overcome the Coulomb force more easily and result in more reactions. Similarly, if the density of the plasma is higher then there will be a greater probability of a reaction occurring for a given time. Thus the reaction rate R_{DT} is given by

$$R_{DT} = n_D n_T \langle \sigma v \rangle = \frac{1}{4} n^2 \langle \sigma v \rangle \tag{1.7}$$

where σ is the ion cross-section and v is the ion velocity, and the fusion reaction rate is given as $\langle \sigma v \rangle$ is averaged over a Maxwellian distribution. The dependence of the reaction rate on the ion temperature for various reactions is shown in Figure 1.1.

To calculate P'_{fus} given in Equation (1.5) it is first necessary to calculate P_{fus} which is merely the reaction rate in Equation (1.7) multiplied by the energy per reaction, E.

$$P_{fus} = R_{DT}E = \frac{1}{4}n^2\langle \sigma v \rangle E \tag{1.8}$$

which represents all 17.6 MeV of energy produced per reaction. However, the 14.1 MeV neutron, being a neutral particle, is not contained and therefore does not contribute to the plasma heating. Only the 3.5 MeV alpha particle is expected to remain and add energy to the system. Therefore, E' represents the energy of the alpha particle and P'_{fus} which represents the power density added to the plasma system from the alpha particle given by

$$P'_{fus} = R_{DT}E' = \frac{1}{4}n^2\langle \sigma v \rangle E'. \tag{1.9}$$

This brings us back to Equation (1.5): if $P'_{fus} + P_{in} > P_{loss}$ then the total energy in the plasma system increases, if $P'_{fus} + P_{in} < P_{loss}$ then the total energy decreases, and if they are equal then the energy of the plasma system is constant. With this in mind, let the *energy confinement time* be τ_E, and the *amplification factor* be Q, also called the *Q-factor*. The energy confinement time is then defined as

$$\tau_E = \frac{W}{P_{loss}} \tag{1.10}$$

and the Q-factor is defined as

$$Q = \frac{P_{fus}}{P_{in}}. \tag{1.11}$$

If the power production of the nuclear fusion reaction is equal to the power injected into the plasma then $Q = 1$ and this is referred to as the *break-even condition*. In the case where the fusion reaction produces so much energy that no power needs to be injected into the plasma then $Q = \infty$ and is

referred to as the *ignition condition*. In this condition $P_{in} = 0$ and the nuclear fusion reaction alone is enough to sustain the plasma energy. The highest Q-factor recorded has been obtained by the Joint European Torus (JET) using a deuterium-tritium fuel in 1997 with $Q = 0.67$. It should be noted that this is the Q-factor as determined strictly by the power entering and leaving the plasma. The energy used in all the systems necessary to maintain the plasma would be used to calculate the *engineering break-even condition* but such a calculation is dependent on the specifics of each device and is seldom referenced. However, knowing that all the supporting systems i.e. cryogenic cooling, control systems, magnetic coils, power generation systems, etc. will eventually need to be supported by the energy harvested from the fusion reaction means that in any real reactor would need $Q \geq 30$. Currently, the International Thermonuclear Experimental Reactor (ITER) is under construction and is expected to achieve $Q \geq 10$.

The conditions necessary for a $Q = 1$ or $Q = \infty$ state are not easily compared from one device to another. High density and low temperature in one device might be better than the higher temperature but lower confinement time in another device. For that reason, the *Lawson Criterion* is used. Assuming the plasma energy remains constant, $\dot{W} = 0$, then it is possible to recalculate Equation (1.5) using the following relationships

$$\begin{aligned} P_{loss} &= \frac{W}{\tau_E} \\ P_{in} &= \frac{P_{fus}}{Q} \\ P'_{fus} &= \frac{E'}{E} P_{fus} \end{aligned}$$

to get

$$\frac{W}{\tau_E} = P_{fus}\left(\frac{E'}{E} + \frac{1}{Q}\right). \tag{1.12}$$

Using Equations Equation (1.2) and Equation (1.8) it is now possible to write Equation (1.12) as

$$n\tau_E = \frac{12 k_B T}{E\left(\frac{E'}{E} + \frac{1}{Q}\right)\langle \sigma v \rangle} \tag{1.13}$$

This is the general expression for the Lawson Criterion. However, for the case of deuterium-tritium ignition the expression becomes

$$n\tau_E \geq \frac{12k_BT}{E'\langle\sigma v\rangle} \tag{1.14}$$

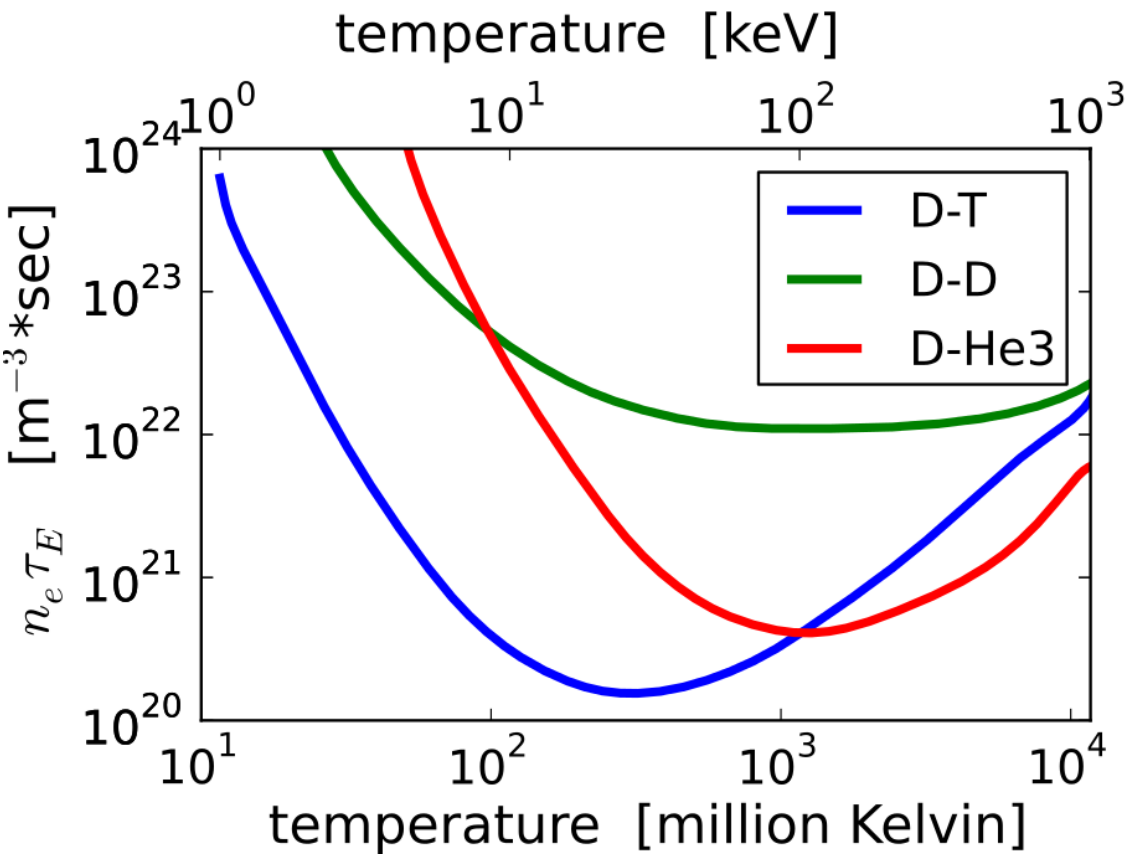
and is shown graphically in Figure 1.2a. Using the minimum of the deuterium-tritium curve in this graph to find the temperature $T = 25$ keV (which is approximately the minimum), the Lawson Criterion becomes $n\tau_E \simeq 1.7 \times 10^{20}$ m^{-3} s.

As mentioned before, the two main confinement methods are inertial and magnetic. Here it is easy to see a major difference between the two: in inertial confinement, the energy confinement time is quite small (10 ps $\leq \tau_E \leq$ 100 ps) and the density is quite high ($\sim 10^{31}$ m^{-3}, more than three orders of magnitude greater than the density of solid hydrogen); conversely, in magnetic confinement, densities are smaller ($\sim 10^{20}$ m^{-3}, significantly less than air density), so the energy confinement time has to be larger ($\sim$ 1 s). This gives good reasoning for using the Lawson Criterion as a figure of merit for different machines. However, the temperature ranges of the various fusion devices can be quite different so an extension of the Lawson Criterion is given by the triple product which includes the temperature of the plasma, shown in Figure 1.2b.

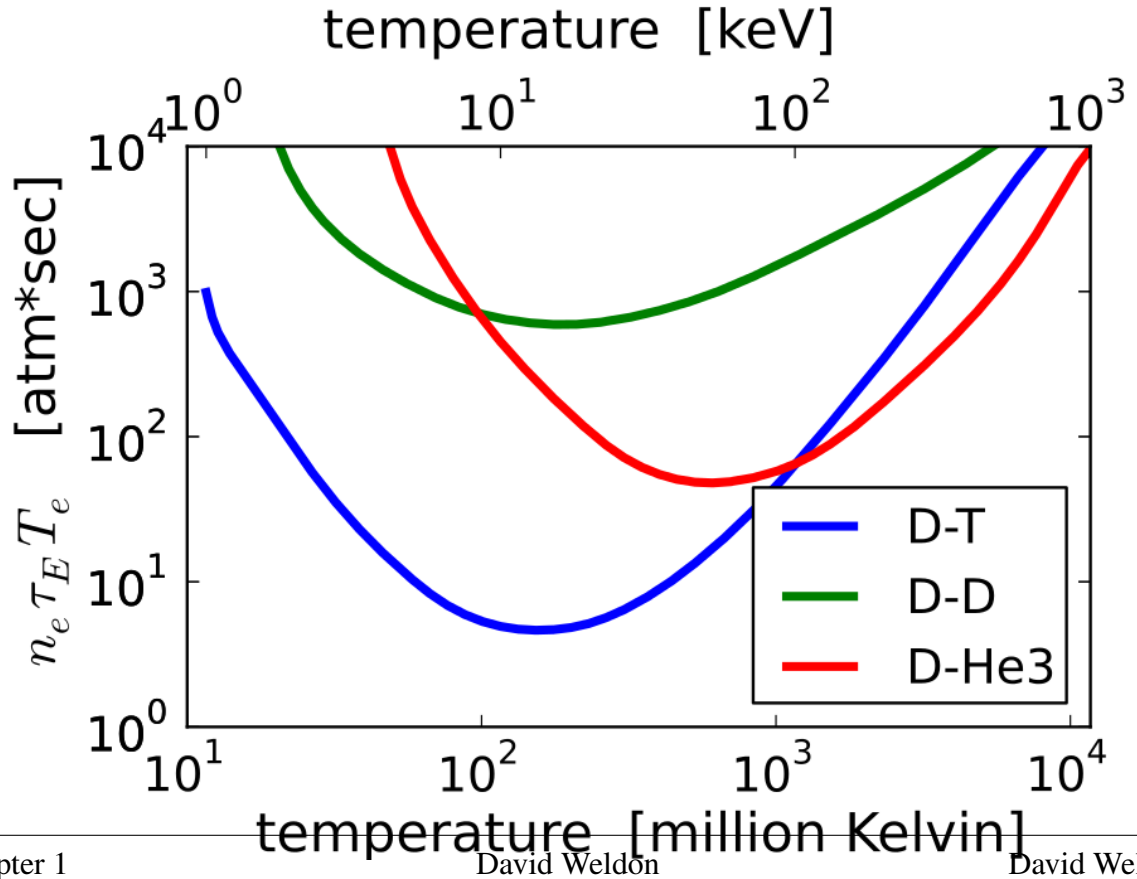
$$nT\tau_E \geq \frac{12k_BT^2}{E'\langle\sigma v\rangle} \tag{1.15}$$

The maximum attainable plasma pressure p is a constant across the different confinement methods, $P_{fus} \propto p^2\langle\sigma v\rangle/T^2$. Therefore, the maximum fusion power attainable, regardless of the confinement method or machine, is temperature T where $\langle\sigma v\rangle/T^2$ is a maximum. Again, using Figure 1.2b to find the minimum of the deuterium-tritium curve gives a temperature $T = 14$ keV (which is approximately the minimum), the triple product then is $nT\tau_E \simeq 3 \times 10^{21}$ keV m^{-3} s. To aid in understanding just how various devices have performed in this regard, Figure 1.3a shows the triple products achieved to date.

To give an idea of how the confinement has progressed over time, Figure 1.3b shows the same achievements as Figure 1.3a but plotted against time so that the progress can be compared to

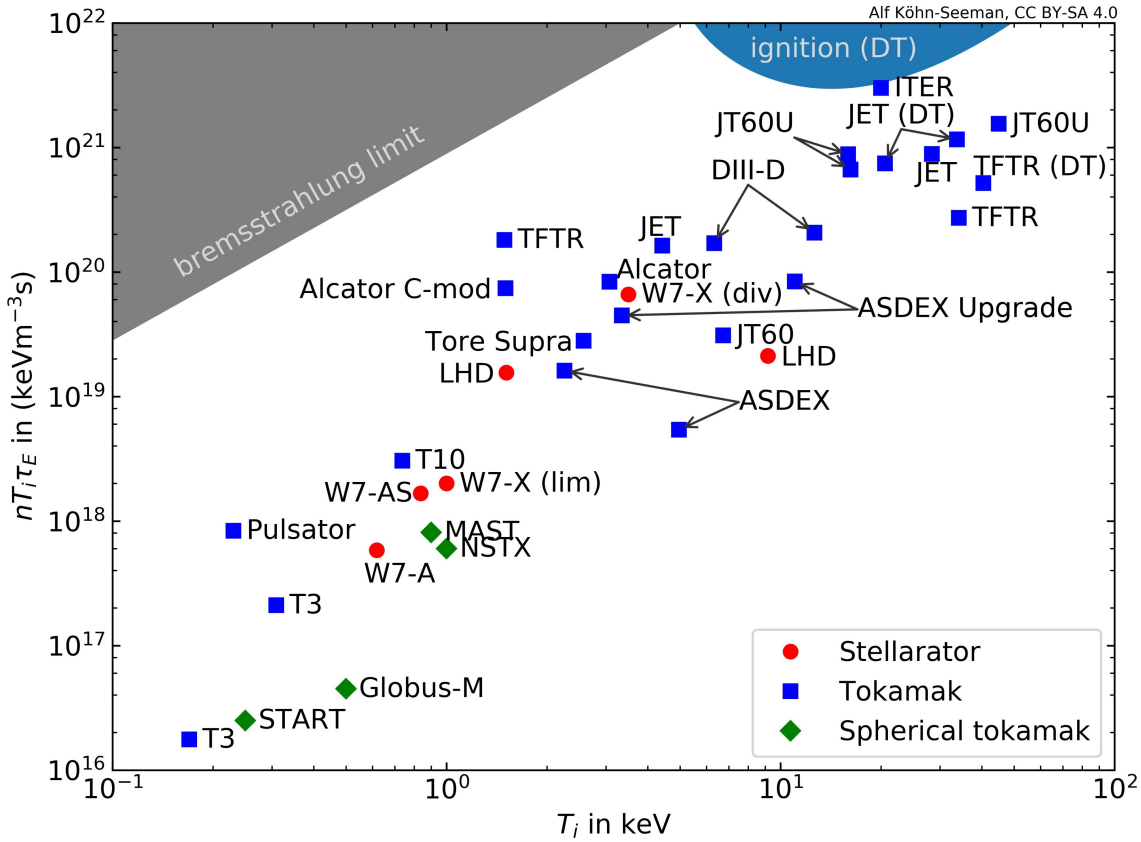


(a) The Lawson Criterion for three common nuclear fusion reactions.

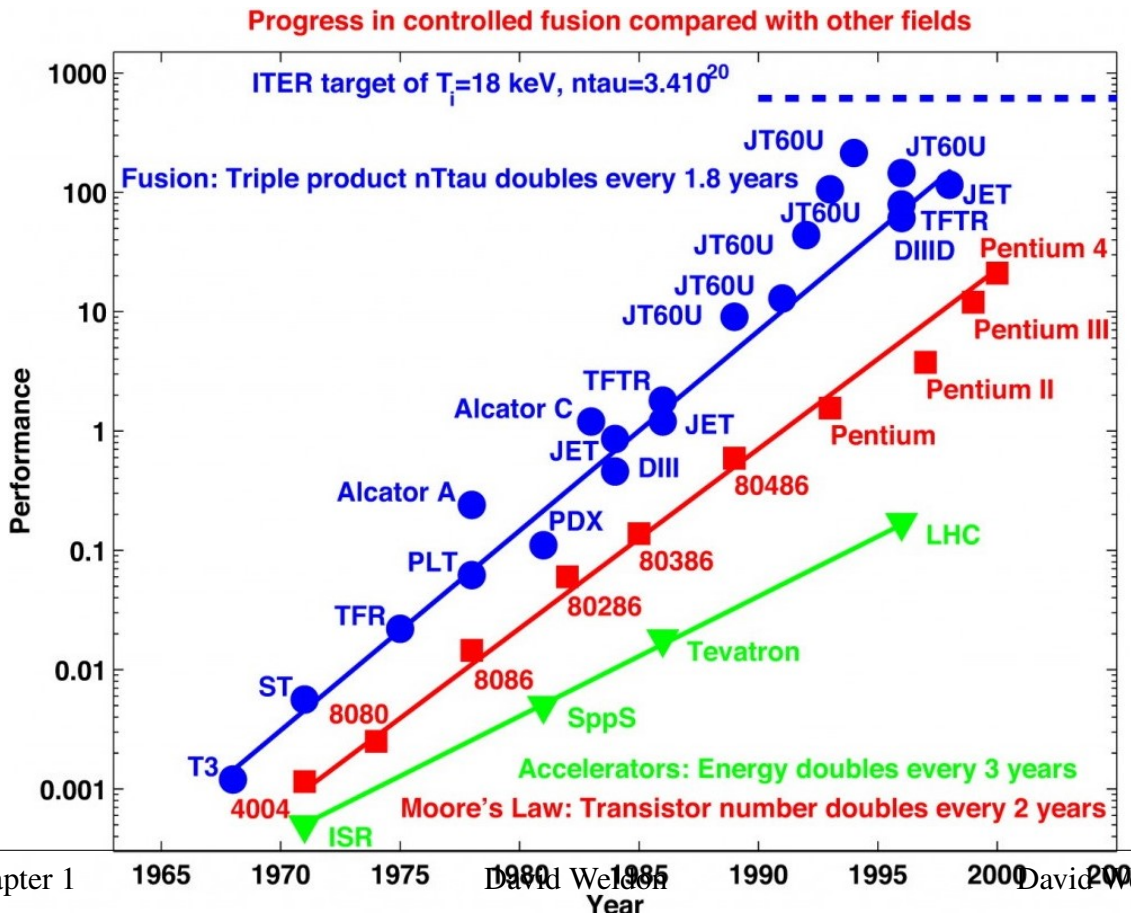


(b) The triple product for three common nuclear fusion reactions.

Figure 1.2: A comparison between the Lawson Criterion and the triple product. By Dstrozzi - Own



(a) The triple products achieved by various magnetic confinement devices from around the world. By Alf Köhn-Seeman, CC BY-SA 4.0



(b) Progress in plasma confinement performance compared to that of other advanced technologies, `https://www.iter.org/newsline/53/1589`

Moore's Law and the achievements of accelerators.

Now that an understanding of the basic criterion needed for fusion energy to be achieved has been established, the next section will focus on the magnetic confinement method. While inertial confinement is still being researched, the fusion community has more or less reached the consensus that magnetic confinement is the more feasible of the two methods for the near future.

1.2 Magnetic confinement

For nearly 90 years now, thousands of researchers have poured millions of hours into developing, refining, and evaluating various methods of using magnetic fields to confine hot plasma well enough to achieve a sustained fusion reaction. Magnetic fields have been relentlessly researched in the field of fusion confinement because of the Lorentz force $\vec{F} = q\vec{E} + q\vec{v} \times \vec{B}$ which tends to keep charged particles trapped in gyroscopic orbits around a particular field-line thus providing a way for the ions to remain very energetic without coming into contact with other materials. This force also results in a characteristic length called the gyro- or Larmor radius expressed as:

$$r_L = \frac{m v_\perp}{|q| B} \tag{1.16}$$

where r_L is the Larmor radius, m is the mass of the particle, $v_\perp$ is the velocity of the particle perpendicular to the magnetic field, $|q|$ is the charge of the particle (the sign of the charge is ignored for the radius but does impact the direction of rotation), and B is the magnitude of the magnetic field.

1.2.1 Magnetic mirrors

To make use of this force one might imagine a solenoid consisting of a series of coils, the basic concept of which is shown in Figure 1.4. The book *Introduction to Plasma Physics and Controlled Fusion* by Chen [8] is an excellent resource and gives a much more detailed explanation of how a

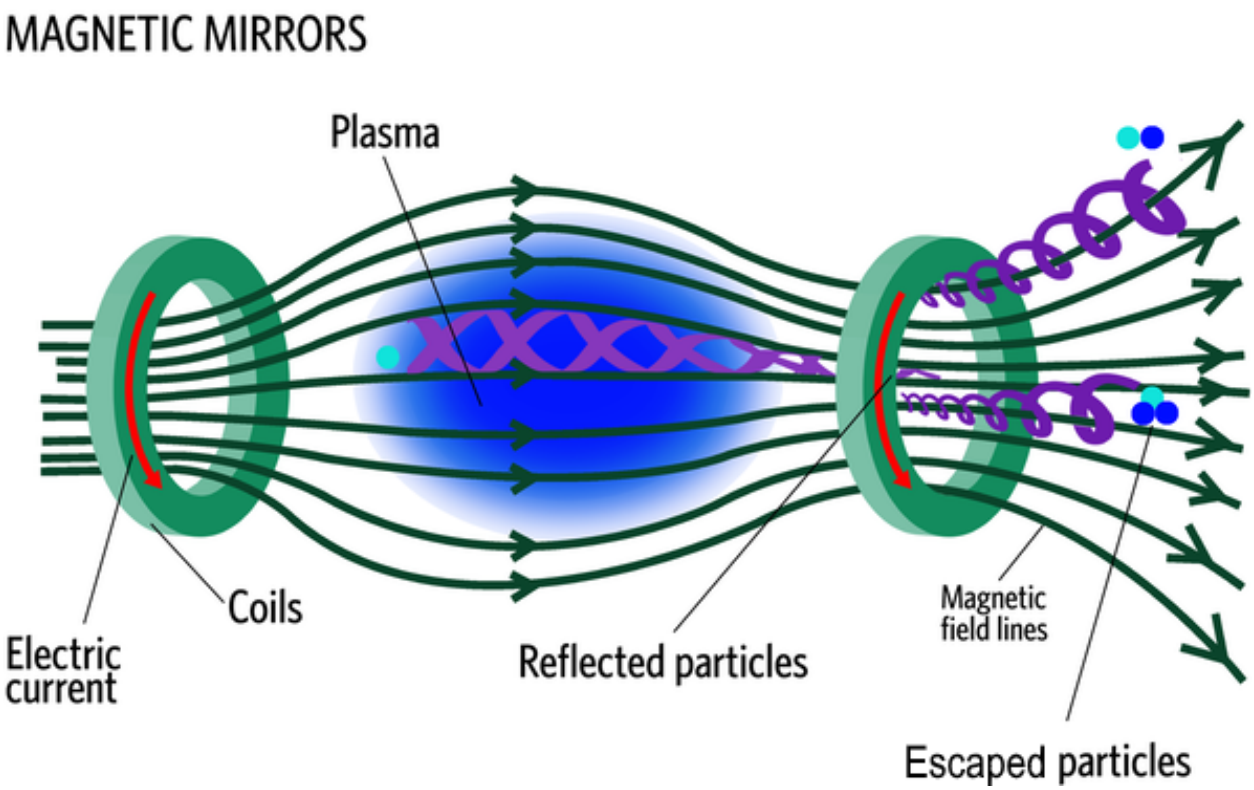


Figure 1.4: With strong magnetic coils on each end, the magnetic mirror confines particles by reflecting them with high magnetic fields. Edited from [7].

magnetic mirror works, but in short it is all due to the dipole moment of the electric current loop that is created any time a charged particle gyrates. This can be expressed as

$$\mu = \frac{\frac{1}{2}mv_{\perp}^2}{B} \tag{1.17}$$

where μ is the dipole moment, m is the mass of the particle, $v_{\perp}$ is the velocity of the particle that is perpendicular to the magnetic field, and B is the magnitude of the magnetic field. This dipole moment is conserved for the time and length scales that are considered in magnetic confinement fusion physics and is referred to as the *first adiabatic invariant*. The other invariant is well known as it is used in many physical systems: energy.

$$\mathcal{E} = q\phi + \frac{1}{2}mv_{\parallel}^2 + \frac{1}{2}mv_{\perp}^2 \tag{1.18}$$

For simplicity, assume the electric potential energy expressed by $q\phi$ is zero because the electric field is assumed to be zero. While the derivation is not detailed here, one can imagine that if the dipole moment is conserved and the total energy is conserved, then when the particle drifts along the field line to either end of the magnetic bottle depicted in Figure 1.4 B increases, then to keep μ constant, $v_{\perp}$ must also increase. But the energy for this increase must come from somewhere, and since this is assumed to be a closed system, that somewhere happens to be the energy in $v_{\parallel}$. Thus, this magnetic bottle can and does do a better job of keeping the particles confined than a uniform magnetic field would. Even so, some particles will escape and so the energy and the particles of the bottle are not sufficiently confined for fusion to occur. How to solve this confinement issue? What if the two ends of the bottle were connected so as to bend the solenoid around into a doughnut shape? Then the escaped particles will just go around in a circle instead of actually escaping. This is exactly the concept that Andrei Sakharov sent to the USSR Academy of Sciences in 1950.

However, this solution adds a complication: the magnetic field is no longer cross-sectionally uniform. This seemingly simple problem with the torus shape has led to more than 70 years of research into solving it. When the coils are all arranged in a straight line the magnetic field can be

very close to uniform throughout the cross-section of the solenoid and so the particles do not drift except along their respective field lines. This is not the case when the solenoid is wrapped around to itself to form the torus. In this geometry, the coils of the solenoid (referred to as the toroidal field coils) will be denser on the inside of the torus than the outside of the torus which results in a high field side and low field side, respectively. This gradient in the magnetic field is responsible for the characteristic $\nabla \boldsymbol{B}$ drift which is present in all toroidal configurations and is shown qualitatively in Figure 1.5 Here, the magnetic field vector $\boldsymbol{B}$ is coming out of the page on the z-axis while the

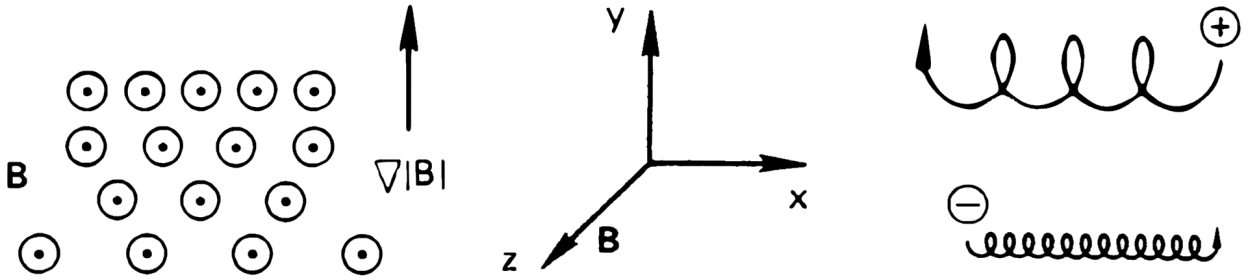


Figure 1.5: Gradient-B drift taken from [8]

strength of the magnetic field increases up the page along the y-axis. This causes the ions and electrons to have a small Larmor radius at the top of their orbit and a larger Larmor radius a the bottom of their orbit causing a net drift in a direction that is perpendicular to both $\boldsymbol{B}$ and $\nabla \boldsymbol{B}$ but because their signs are opposite, their direction of rotation is also opposite which means their drift directions are opposite as well. This causes yet another issue as the charge separation will cause a non-zero $\boldsymbol{E}$ field which will also induce a drift on the particles. This $\nabla \boldsymbol{B}$ is also accompanied by a drift due to the curvature of the magnetic field. As the particles drift along the magnetic field lines they must experience a centrifugal force outward as shown in Figure 1.6. This force also causes a drift that is perpendicular to the force and the magnetic field. Unfortunately, this drift is always in the same direction as the $\nabla \boldsymbol{B}$ drift, exacerbating the problem. What started as a simple solution to the problems of the magnetic bottle has clearly spawn like hydra into a myriad of other problems with particle confinement.

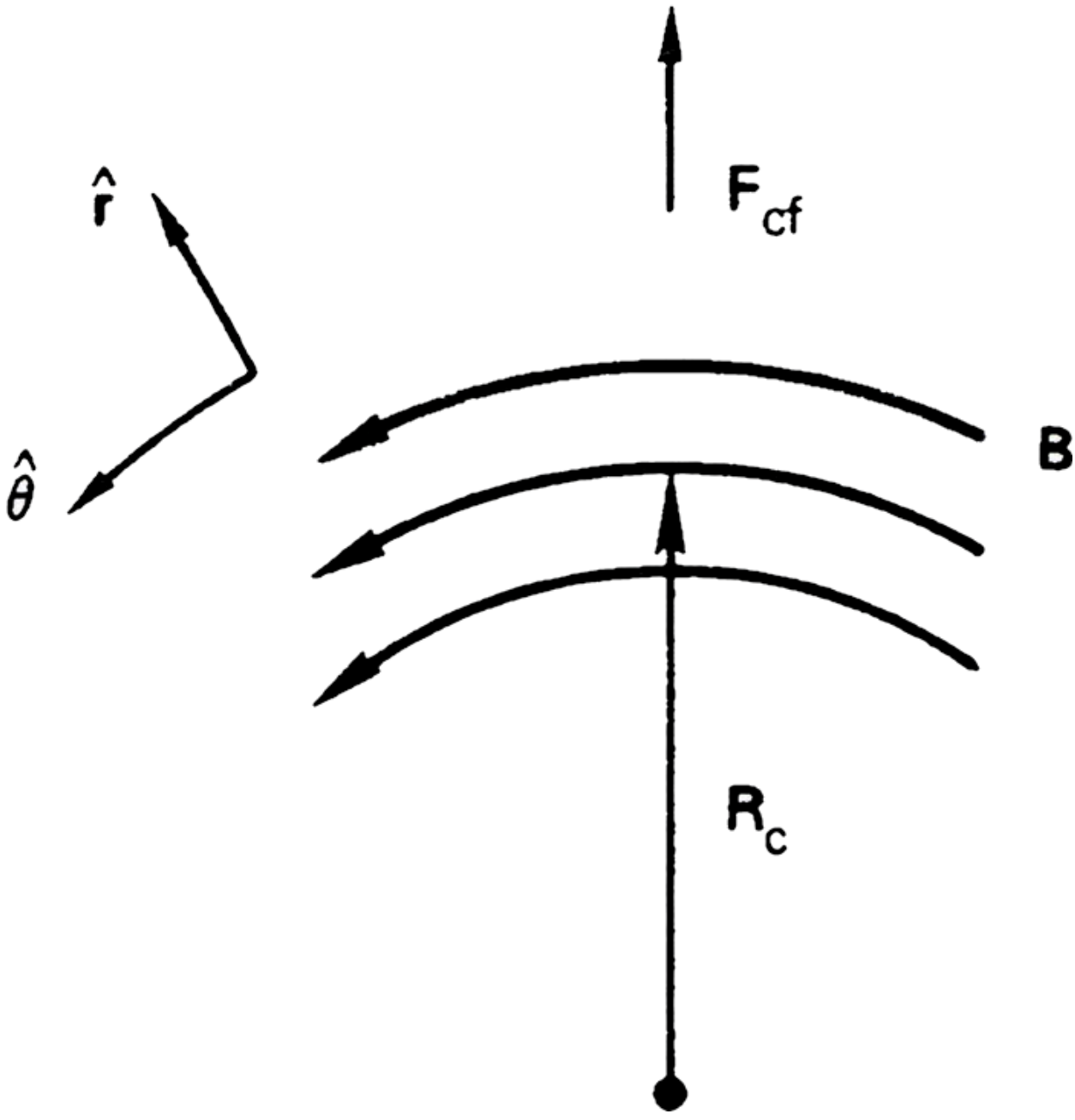
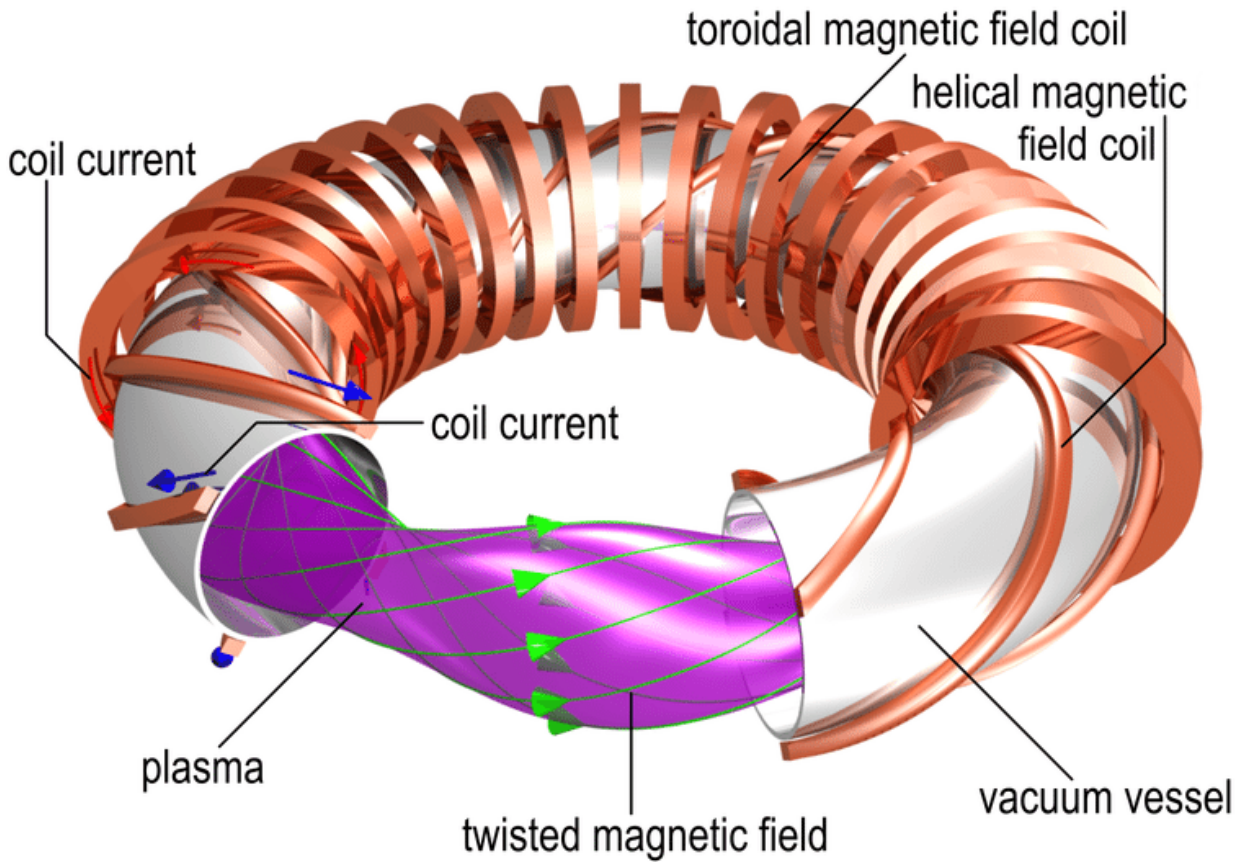


Figure 1.6: Curvature drift taken from [8]

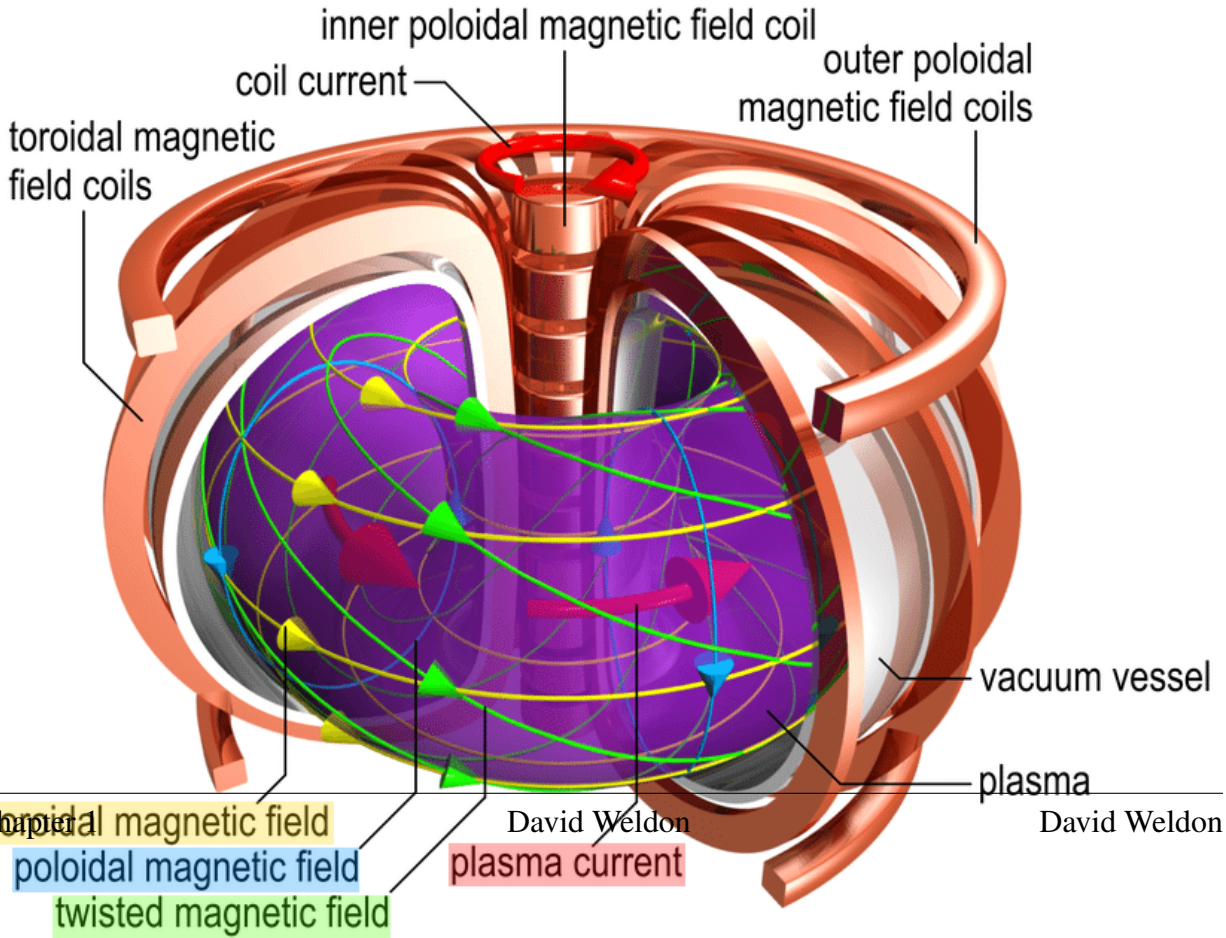
1.2.2 Tokamaks

There are two main branches of magnetically confined plasmas: the stellarator and the tokamak, Figure 1.7. Of the magnetic confinement schemes, the most well researched is the tokamak which is a Russian abbreviation for "toroidal chamber with magnetic coils" (from the transliteration of the Russian sentence toroidal'naya kamera s aksial'nym magnitnym polem or toroidal chamber with an axial magnetic field) and is displayed in an artistic rendering in Figure 1.7b. However, the stellarator design is gaining in popularity thanks to the great advances made in computer design capabilities and material science. This is because the geometry for the helical magnetic field coils shown in Figure 1.7a is non-trivial, to say the least, but with computers to perform the incredibly difficult calculations necessary for such complex geometry, this is less of an obstacle. In the same way, winding a continuous helical magnetic field coil around the vacuum vessel would be extremely difficult during the construction of a stellarator. A solution is to fabricate the coil in sections and then connect them in place. Previously this posed a problem as the connections would have unacceptably high resistivity. The heat dissipation would put a limit on the current and consequently the magnetic fields. Fortunately, material science has advanced to the point that these connections could be superconducting and heat dissipation can be overcome. Despite these recent advances, however, the tokamak design is still the more mature of the two and is the focus of this research though future applications to the stellarator are not improbable.

As seen in Figure 1.7b, the tokamak is a solenoid with its ends connected thus making the torus shape which results in the magnetic field created by the solenoid to be continuous around the torus which is referred to as the *toroidal magnetic field* in the figure. When the solenoid is closed in the torus shape, the coils of the solenoid are referred to as the *toroidal magnetic field coils* in the figure (also referred to as the Toroidal Field or TF coils). However, the toroidal field alone is not enough to keep the plasma stable because of the inherent drifts as discussed in Section 1.2.1. To combat this inherent instability, a *poloidal magnetic field* as labeled in the figure (also referred to as the Poloidal Field, PF), is introduced via a *plasma current* that flows in the toroidal direction. To induce this plasma current, typically an *inner poloidal magnetic field coil* is introduced in the



(a) Artistic rendering of a stellarator



center of the torus (through the doughnut hole) that acts as the primary side of a transformer and is referred to as the central solenoid. While there are other schemes to drive the current in the plasma such as neutral beams, a phenomenon known as *bootstrap-current*, and EM wave interactions, this transformer method is still the primary source of plasma current in most tokamaks. The *outer poloidal magnetic field coils* can also be used to drive the plasma current, but are most often used for plasma shaping, control, and stability. This, of course, leads to an inherent pulsed operation of the tokamak as no transformer can have a continuously increasing current and running a sinusoidal waveform through the central solenoid would cause the plasma to die at each peak or trough of the input current in large plasmas and therefore a near-total confinement loss. The superposition of the toroidal and poloidal magnetic fields causes the resulting field lines to curve around the torus in a helical path, shown in Figure 1.7b as the *twisted magnetic field*. These helical field lines of the same flux create surfaces that are nested something like the layers of an onion and are referred to as flux surfaces, in the figure only a single representative surface is shown. While this scheme is effective in creating quasi-stable plasmas, it is not generally sufficient for shaping and controlling the plasma. So, as stated above, additional *outer poloidal magnetic field coils* referred to as the poloidal field coils (PF coils) are added both inside and outside the vacuum vessel in which the plasma is produced.

The nested flux surfaces discussed above are generally able to contain a plasma ion for 0.1 - 1.0 seconds [10], which is far from sufficient for fusion energy production as discussed in Section 1.1.3. Diffusion, drift, and turbulence in the plasma cause the energetic ions to escape beyond the Last Closed Flux Surface (LCFS) and eventually hit a Plasma Facing Component (PFC) while tracing an open field line (a magnetic field line that passes through a PFC). Typically, the ions that escape in this way are not as energetic as those still trapped in the center of the plasma, but they still have temperatures on the order of hundreds of eV. This not only damages the PFCs but also causes substantial amounts of energy loss from the plasma. This leaves ample room for a plethora of schemes for improving *plasma confinement*. This section builds upon the explanation in Section 1.1.3 of plasma energy confinement time, τ_E, and narrows it to mean confining energy

and particles within the LCFS of the plasma. In general, anything outside the LCFS of the plasma is considered to not be in the plasma system insofar as the energy balance is concerned. Figure 1.8 shows the LCFS and the vacuum vessel in 2D and 3D for the Experimental Advanced Super Conducting Tokamak (EAST), the main tokamak of this work.

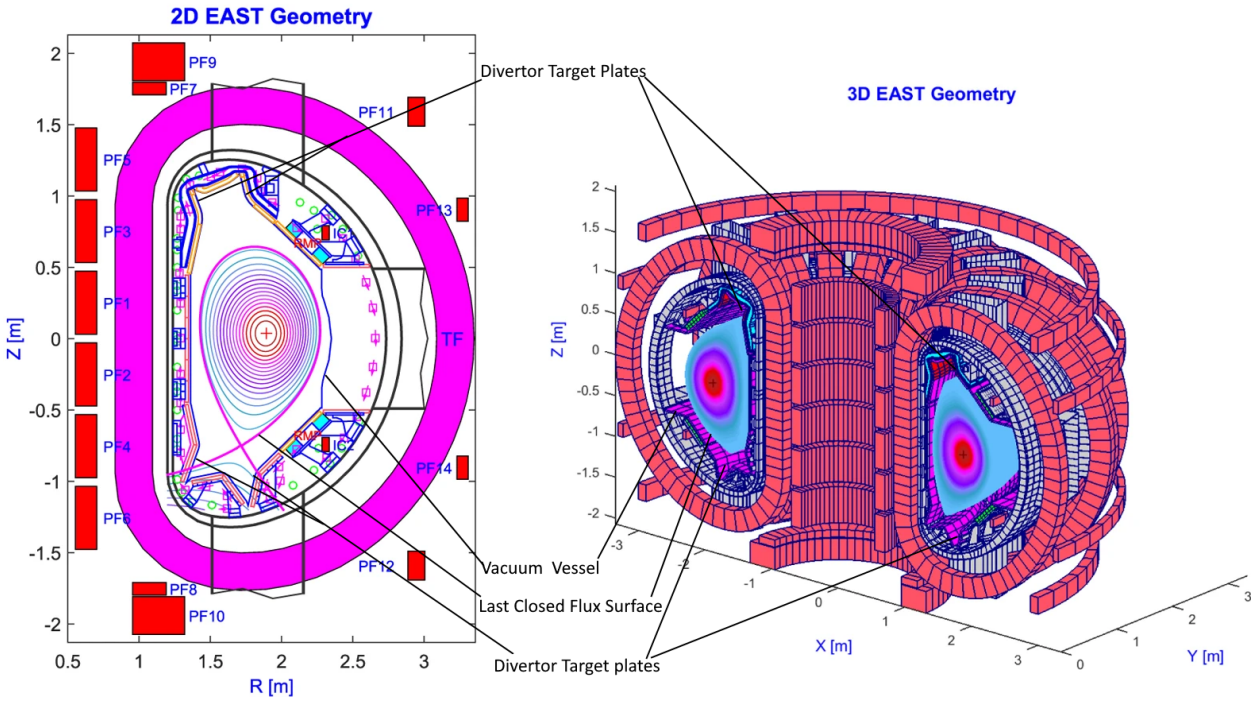


Figure 1.8: The geometry of the EAST device: the 2D view in the poloidal plane is on the left and the 3D view of one half of the tokamak is one the right. PF represents the number of Poloidal Field Coils while TF refers to the number of Toroidal Field coils. [11]

Because the temperature, pressure, and density are not uniform throughout the plasma, various kinds of transport move energy and particles out of the plasma. Most of these transport methods are due to turbulence, which is the same turbulence seen in ordinary fluid hydrodynamics. This violent and chaotic motion of the plasma can cause instabilities in the plasma that can increase radiation losses, eject particles and energy to the Scrape-Off Layer (SOL), and eventually out of the plasma, or cause magnetic field line disconnection. The last of which is the most violent and energetic of the three transport modes and can easily release enough energy from the plasma to damage the vacuum vessel and cause the plasma to cool to the point that it is lost. The specifics of each transport mode will not be discussed in this thesis, only note that non-homogeneity leads to

instabilities which lead to transport [8].

1.2.3 Tokamak geometry

Before going deeper into tokamak physics, it is best to introduce some more torus and tokamak geometry. Figure 1.9 shows the Z, φ, and ϑ, directions. These are the coordinates most commonly used in toroidal geometries. Sometimes ζ will be used interchangeably with φ depending upon the author. As shown in the figure, Z is the vertical direction, φ is the toroidal direction, and ϑ is the poloidal direction. R is the major radius coordinate of the torus, R_0 is the distance to the geometric center of the poloidal cross-section, a is the minor radius, and r is the minor radius coordinate. In general, tokamaks are considered to be toroidally symmetric. That is to say, the plasma parameters are not functions of φ. This, however, is not true for stellarators, though they do have toroidal periodicity.

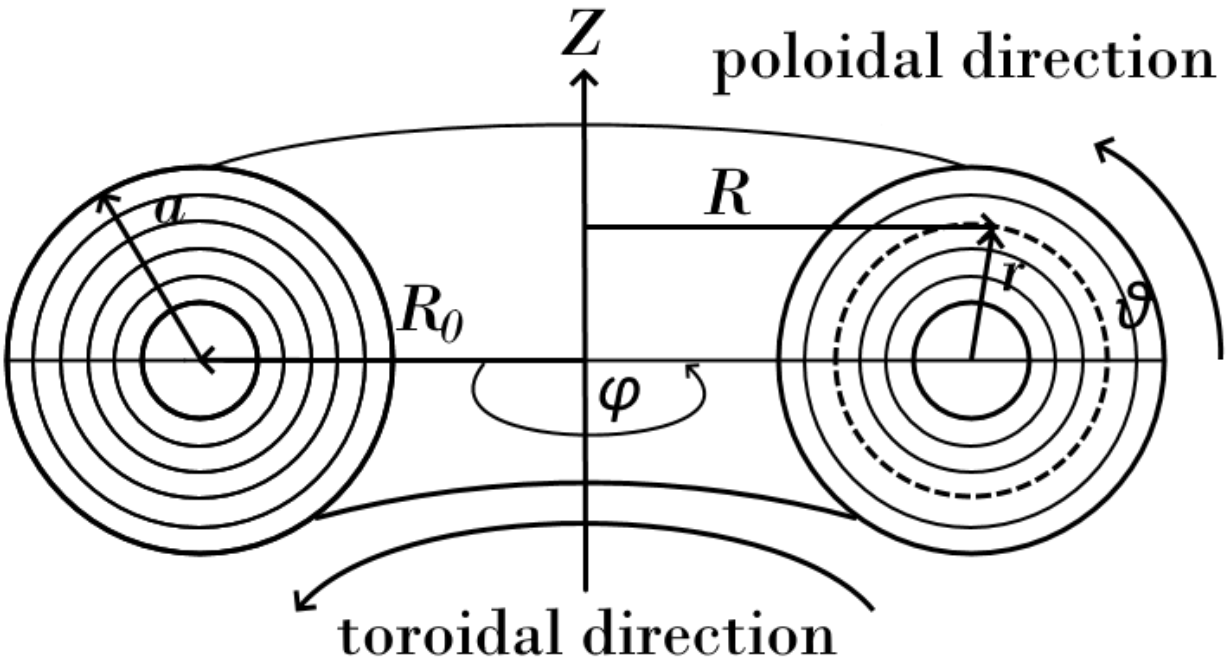


Figure 1.9: The geometry of a torus with a circular cross-section. This is also the geometry of the most basic stellarator or tokamak plasma. [12]

Figure 1.9 shows the most basic shape of a plasma in a tokamak. This is from the first fusion reactor designs from the 1950s mostly because they were easier to design and understand. In the

1960s, the inner poloidal magnetic field coils, shown in Figure 1.7b, were used to create the plasma current and poloidal magnetic fields to suppress instabilities and control the plasma position more precisely. By the 1980s researchers at several devices had demonstrated that the outer poloidal magnetic field coils in Figure 1.7b could make the twisted magnetic lines not just helical, but also non-symmetric. In general, a non-symmetric plasma is referred to as a *shaped* plasma and as research continued greater and greater advances in confinement and stability were made using highly shaped plasma [6].

This shaping eventually led to the *diverted plasma* as opposed to a "limited plasma". A limited plasma has its LCFS touching the limiter (a protrusion from the vacuum vessel wall into the plasma area to prevent the plasma from coming into contact with more sensitive parts of the wall and to fix the plasma potential, temperature, and position) while a diverted plasma has a *separatrix* and an *X-point* in the magnetic field allowing the LCFS to be aimed at specific divertor plates designed to handle the energy of the particles traveling on the incident field lines. This is better understood by looking at Figures Figure 1.8 and Figure 1.10; the separatrix is the transition region from closed field lines to open field lines. Inside of the separatrix is the confined plasma and outside the separatrix is the SOL. By changing the current in one or more of the PF coils, one or more X-points can be created. An X-point is any place where the magnitude of the poloidal magnetic field becomes null. Therefore, the separatrix is the magnetic flux surface that intersects an X-point and since any magnetic flux surface that is outside this separatrix is not confined (i.e., open magnetic field lines), the separatrix defines the LCFS. In short, limited and diverted plasmas have an LCFS but only diverted plasmas have a separatrix, for this thesis. In limited plasmas, particles that escape the LCFS merely follow the field line until they strike an object which could potentially damage diagnostic instruments or erode the vacuum vessel. But in diverted plasmas, the X-point and separatrix cause the plasma edge to decouple from the limiter thus giving the escaped particles a preferred path and an established area to strike. This area is commonly known as the divertor target plate or divertor target [6].

Shaping a plasma naturally calls for a way to characterize the shape, thus two new parameters

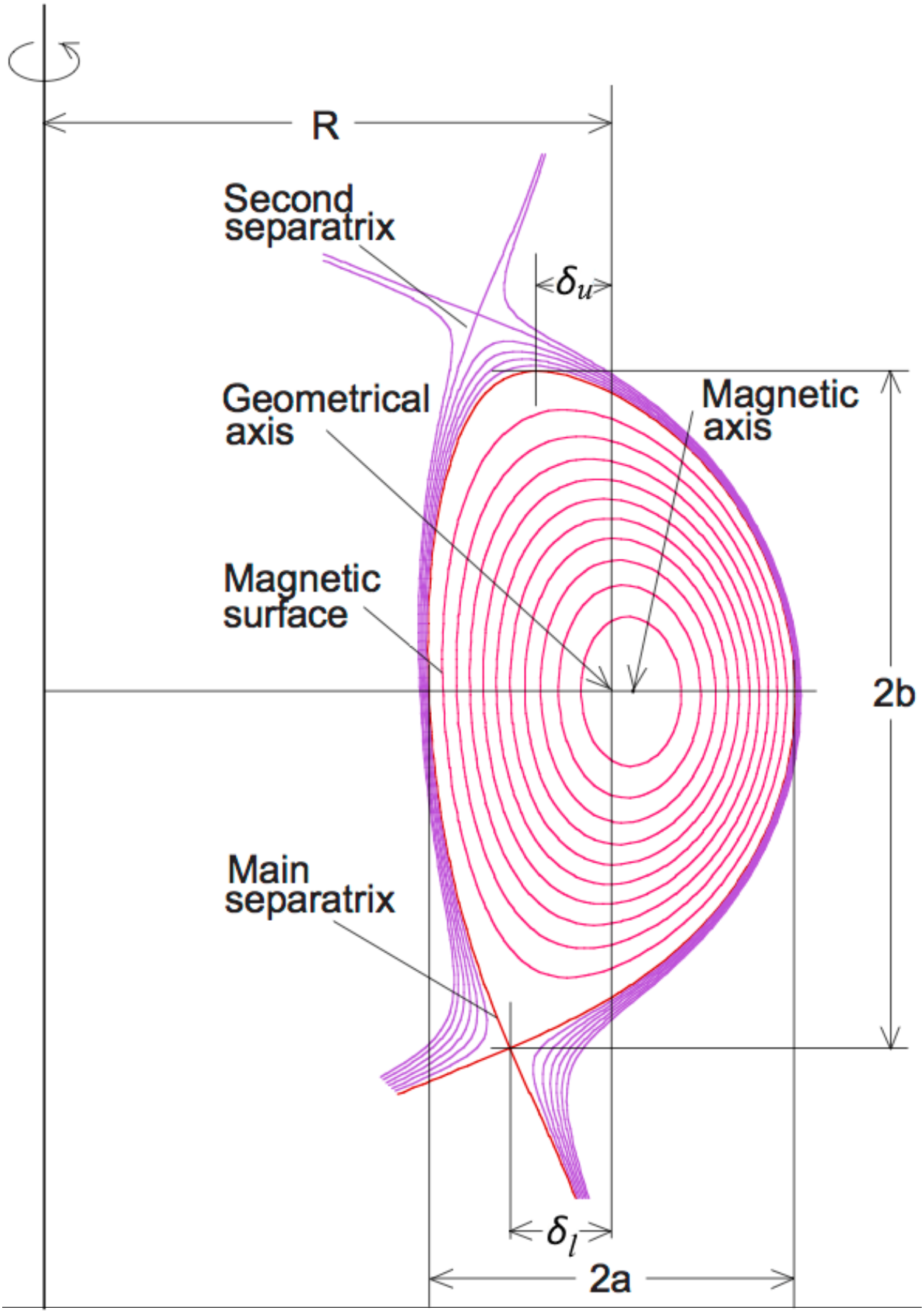


Figure 1.10: The geometry of a shaped plasma. Modified from [13]

are introduced: elongation and triangularity. Elongation is a ratio of the vertical dimension of the plasma to the horizontal dimension of the plasma, formally calculated as $\kappa = b/a$ where b is the height of the plasma above the geometric mid-plane of the torus. The triangularity is defined by

$$\delta_U = \frac{\left(R_{geo} - R_{upper}\right)}{a} \tag{1.19}$$

$$\delta_L = \frac{\left(R_{geo} - R_{lower}\right)}{a} \tag{1.20}$$

Where R_{geo} is the radius to the geometric center of the plasma, R_{upper} and R_{lower} are the radii to the vertical maximum and minimum of the LCFS, respectively, and finally, δ_U and δ_L are the upper and lower triangularities, respectively. Note that in Figure 1.10 the vertical minimum is also the X-point; this is not always the case that the X-point is a vertical extreme, but is most often the case. In general, the greater the elongation, the more difficult it is to control the vertical stability of the plasma [6]. Finally, the triangularity of the plasma plays a significant role in the stability and confinement of the plasma. This improvement is so great that nearly all large tokamaks and stellarators use a D-shape plasma as a standard. Note that in Equation (1.19) and Equation (1.20) if the $R_{upper/lower} > R_{geo}$ then the triangularity could be negative. If both are negative, then the plasma would have something of a reversed D-shape [6]. The remainder of this thesis will address a revived area of interest in regards to plasma shape: the negative-triangularity configuration.

1.2.4 Negative-triangularity

As previously stated, all tokamak experiments have only achieved pulsed operation, however, there is significant research being done in plasma current driving techniques [14] as well as in an intriguing phenomenon known as bootstrap-current [15]. While it now seems that the current drive challenge of continuous operation will be overcome, this is not the only challenge facing continuous operation. In recent years, tokamak research has repeatedly shown that the plasma edge (that is the layer closest to and including the LCFS) magneto-hydrodynamic stability is critical for handling the power to the vacuum vessel walls and the divertor target plates which is now, and will

most likely continue to be, a limiting factor in the ITER and the DEMOnstration Power Station (DEMO). The heat flux is not only critical during Edge Localized Mode (ELM) energy emission (a type of violent transport caused by turbulence that can be mitigated to the point that they do not terminate the plasma) but also between ELM emissions. Because the SOL heat channel is so narrow, the divertor target plate must sustain a very high steady-state heat flux on a relatively small area. Both the electron conductive heat flux and the ion convective heat flux are quite large due to the enhanced SOL flow, which causes the heat channel to be so narrow [16]. It has been proposed that neoclassical SOL flow acceleration mechanisms could be the cause of the high SOL flow speed [17]. At the same time, others have suggested that a tokamak plasma configured with a strong negative-triangularity could significantly reduce the SOL flow acceleration due to trapped particles and thereby reduce the heat-load on the divertor [16]. Furthermore, tokamak configuration has evolved to optimize the core plasma confinement which has lead to the D-shaped plasma. In H-mode operation (high energy confinement mode; as opposed to L-mode, low energy confinement mode), this shape gives a high plasma edge pressure limit (located at the LCFS) and reduces edge transport [18]. Again, power handling is now a significant problem for the D-shaped plasma and needs to be addressed before the continuous operation in a tokamak is possible. The majority of Negative-Triangularity Configuration (NTC) experiments have been performed on TCV [19]–[23] and DIII-D [24]; from them, four major conclusions stand out:

1. The heat-load on the PFCs can be greatly reduced
2. The thresholds for certain instabilities (e.g., Mercier, kink, ballooning modes) are higher than for similar Positive Triangularity Configurations (PTC)
3. The vertical growth rate increases sharply as the triangularity becomes more negative
4. The NTC has yet to be established using ITER-like configurations i.e., superconducting, D-shape, target-diverted plasmas

The first two of these three conclusions are the reasons for further study of the NTC while the third is a challenge to be overcome and is currently being addressed [22]. This thesis addresses

the final point from above and presents the first NTC successfully run on EAST which uses the ITER-like conditions of superconducting coils, D-shape vessel, and the plasma diverted onto target plates.

1.3 Thesis summary and structure

This thesis work was originally intended to compare plasma parameters such as vertical growth rate, instability mitigation, and divertor heat-load between a series of positive-triangularity plasma configurations with $0 \leq \delta_L \leq 0.4$ and an equivalent series of negative-triangularity plasma configurations with $-0.4 \leq \delta_L \leq 0$ on EAST. However, after several failed attempts to produce a stable negative-triangularity plasma, it became apparent that since this would be the first time performing a negative-triangularity discharge on EAST, the time needed to configure the plasma control system would be significantly greater than initially expected. To pivot around this setback and make the best use of the allotted experiment time, our team changed our goal: achieving a plasma configuration with the triangularity as negative as possible and then create an equivalent positive triangularity configuration. In conjunction with this would be an engineering analysis of the limiting factors of the EAST device that do not allow for greater negative-triangularity. The negative-triangularity configuration was eventually achieved after significant trial-and-error and was immediately followed by a shutdown of EAST (scheduled maintenance and preparation for upgrades to the vessel wall). Thus, no equivalent positive-triangularity configuration could be achieved for comparison.

This setback was taken in stride and the aim of this research was once again pivoted in the direction of simulation and to prepare teaching tools for the lucky graduate student who would pick up the negative-triangularity torch, so to speak, and soldier on where we had left off. Concerning simulation, a set of `Matlab` scripts used for simulation, design, and analysis have been developed by General Atomics in San Diego, CA, and were adapted for use at EAST and were successfully used to design the negative-triangularity configuration that eventually was achieved at EAST.

Furthermore, the positive-triangularity equivalent was successfully designed with these tools as well, but a comparison to the experimental data from the negative-triangularity configuration is not fruitful as the design does not give the important characteristics such as growth rate, heat-load, etc. However, when discharges at the EAST facility resume, the PTC design can be tested. Finally, the steps for using these tools have been detailed for instructional purposes for a successor.

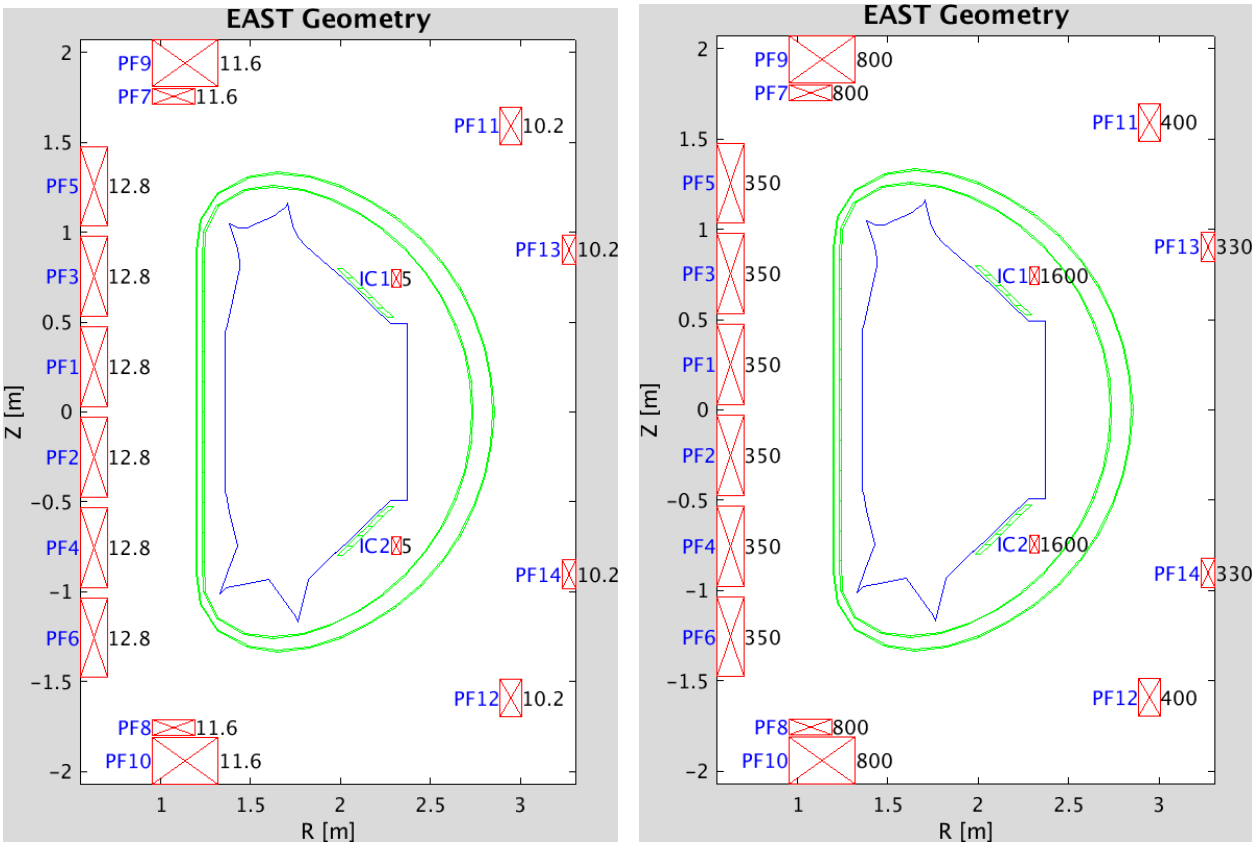
Chapter 2 gives a brief description of EAST and how its geometry and parameters compare with those of TCV and DIII-D followed by a detailed account of the design methods used for the NTC and the specific parameters used in the Plasma Control System (PCS) to achieve a full discharge NTC plasma. Chapter 3 presents the detailed results of the experiment, compares them to the design, and examines the limitations that EAST has in achieving the NTC as compared to the PTC. Chapter 4 gives a detailed demonstration of the simulation software used to model the plasma response and how it is used to fine-tune PTC discharges. Chapter 5 concludes the work done in this thesis and discusses its implication for future work for furthering NTCs at EAST.

Chapter 2

Design & Optimization

2.1 Experimental Superconducting Tokamak (EAST) setup

EAST is a key research project for the Chinese Academy of Sciences (CAS) located in Anhui Province, Hefei city. As shown in Figure 2.2a, EAST has a D-shape vacuum vessel with 12 independently controlled superconducting coils surrounding the vessel. In Figure 2.2a it can be seen that coils 7/9 and 8/10 appear conjoined; each pair is in fact wired in series. EAST is normally operated with $1.5 \leq \kappa \leq 2.0$, $0.3 \leq \delta \leq 0.6$, coil currents below 14 kA as shown in Figure 2.1a, power supplies below 1.1 kV as seen in Figure 2.1b, and cooled by supercritical helium at 4 K. For PTCs the 12 coils and their power supplies have more than enough flexibility to operate in these ranges allowing EAST to explore ITER relevant issues such as particle handling, plasma wall interactions, and non-inductive current drive [25]. However, EAST was commissioned in 2006, well before NTCs became established as an area of interest for ITER and other future tokamak operation. Nonetheless, it is worth the effort to explore the NTC on EAST as the results will help us to investigate engineering solutions to the shaping limitations and the diagnostic data can help in understanding the stability and confinement of an NTC better. It is useful to compare the NTC used on EAST with those of TCV and DIII-D; first look at the important parameters of each device. However, since this research is done with the future tokamaks and fusion reactors in mind, ITER's



(a) Maximum coil current [kA] (b) Maximum power supply voltage [V]

Figure 2.1: Configuration and limits of the PF coil currents and power supply voltage on EAST.

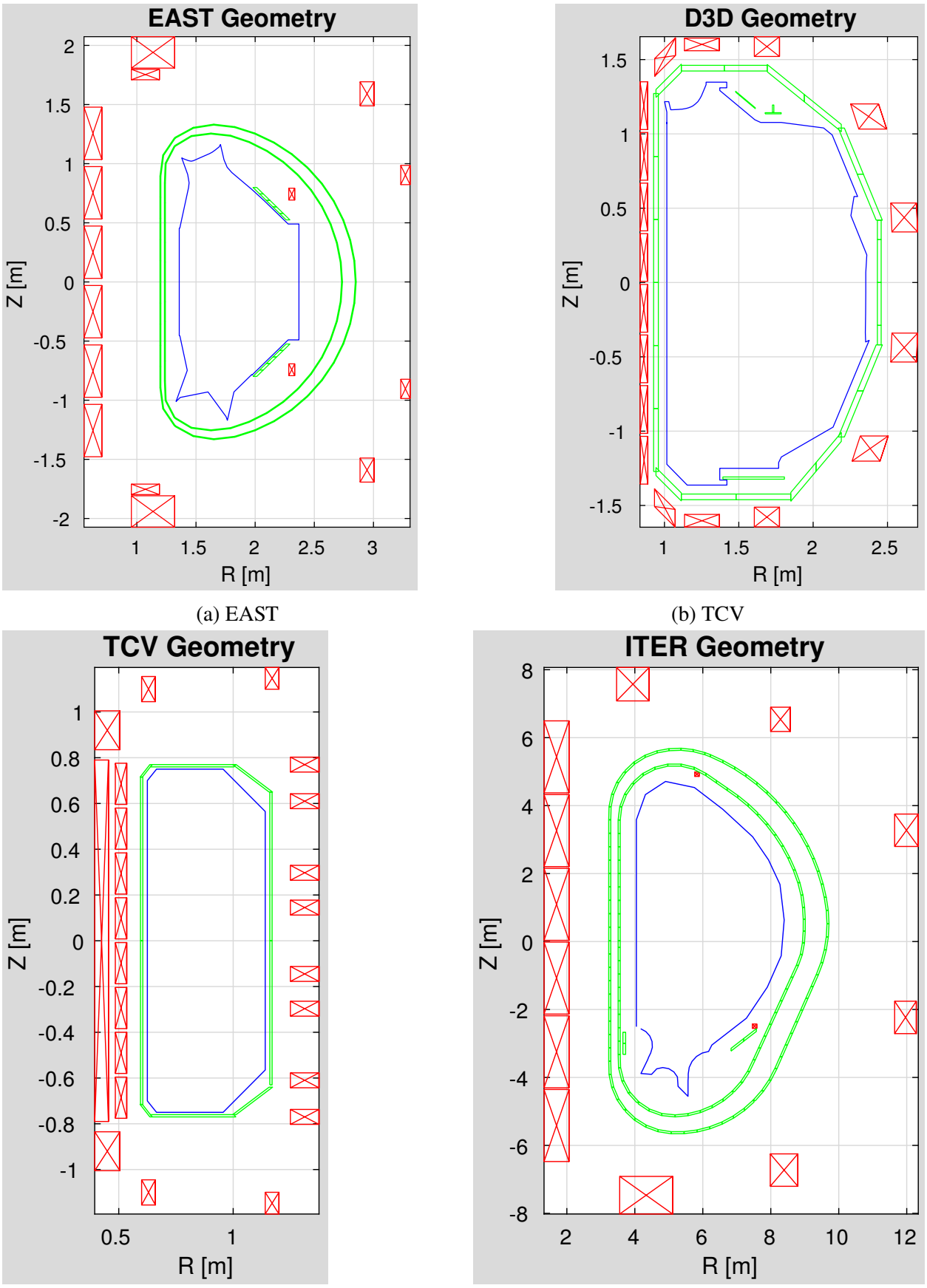
parameters are also included for comparison. Table 2.1 lists the parameters while Figure 2.2 shows the poloidal view of each tokamak.

Table 2.1: Comparison of tokamak parameters for EAST, TCV, DIII-D, and ITER

Parameter	Symbol	Unit	EAST	TCV	DIII-D	ITER
Toroidal Field	B_t	[T]	3.5	1.43	2.2	5.3
Plasma Current	I_p	[MA]	1.0	1.2	2.0	15
Major Radius	R_0	[m]	1.85	0.88	1.67	6.2
Minor Radius	a	[m]	0.45	0.25	0.67	2.0
Pulse Length		[s]	1000	2	7	400
Additional Heating		[MW]	7	4.5	23	50
PF Coils*			12	14	18	12
Shape			D	Rect.	D	D

* Number of independently controlled poloidal field coils

As seen in Figure 2.2b the geometry of TCV allows it to shape plasma in exotic ways and is one of the main reasons TCV has been capable of achieving $\delta \leq -0.9$ which have shown promising results with upper, lower, and both triangularities in such a negative configuration [23]. While this success has been encouraging and continues to provide insights into the stability, confinement, etc., all of the NTC plasmas on TCV have either been non-diverted or diverted onto the vessel wall [20], [26], [27]. This, of course, is not ideal for future fusion devices which will need the plasma diverted safely onto target plates. In contrast, DIII-D as seen in Figure 2.2c is a D-shaped tokamak and has parameters more similar to what future fusion reactors will likely have. However, while DIII-D has divertor target plates, the NTC experiments performed so far have not diverted the plasma onto the targets but instead onto the vessel walls which limits how long a pulse can last before the walls are damaged too much [24]. Even with 18 poloidal field coils, it is extremely difficult to shape the plasma with strong negative-triangularity ($\delta \leq -0.4$) while also maintaining target-diverted plasma. Furthermore, neither of these tokamaks are superconducting which is a key aspect of future fusion devices and presents its own set of advantages and disadvantages when achieving NTC plasmas, namely stronger magnetic fields but also coils farther from the plasma as is evident in Figures Figure 2.2c and Figure 2.2a. The coils on EAST are nearly twice as far from the plasma as those of DIII-D and those of ITER in Figure 2.2d will be even farther. To this end,



(a) EAST

(b) TCV

(c) DIII-D

(d) ITER

Figure 22: Side-by-side comparison of EAST, TCV, DIII-D, and ITER geometries in MEQ. The horizontal axis is the major radius, R, and the vertical axis is height, Z. Note the scales of each.

EAST has successfully achieved target-diverted plasmas with a lower triangularity of $\delta \leq -0.09$ for more than 6 s.

2.2 Design tools: `gsdesign.m`

Previously at EAST, Equilibrium FITing, (EFIT) was the tool to use for both reconstructing the plasma after a shot and for designing a future shot. The PCS has some tools for designing future shots as well, though they are more limited and less convenient. However, all of the design work reported in this thesis was done with a set of `MatLab` scripts called TOKSYS which are maintained by General Atomics of San Diego, California and were largely developed there as well, though scientists and engineers from EAST, KSTAR, ASDEX-Upgrade, and other experiments have also contributed substantially over the years. While TOKSYS is a large and very versatile suite of `Matlab` scripts, functions, and `Simulink` models, consisting largely of a series of circuit models with all of the passive conductors, active coils, and plasma circuit as well as the plasma response model [25], this chapter focuses on the use of just one main script: `gsdesign.m`.

The script is designed to find a numeric solution to the Grad-Shafranov equation while simultaneously minimizing a cost function of several design parameters such as boundary points on the separatrix/LCFS (r,z), coil currents (IC), flux (ψ_p, ψ_b), plasma current (I_p), betas (β_p, β_t, β_n), and many more. While `gsdesign.m` can accept many different file inputs, for this section all of the inputs come from running the standard startup scripts for TOKSYS as well as loading a structure that contains all the equilibrium data from either an EFIT gfile or a previous `gsdesign.m` equilibrium.

Note that for this thesis the words *shot* and *discharge* will be used interchangeably. Both refer to a single attempt of the EAST machine to produce a plasma, whether successful or not.

2.3 Necessary background

To best understand the work presented in the remainder of this chapter, it is imperative that the user first be familiar with `Matlab` (some short cuts are given in Appendix D) and it's general operation. It is also useful to have read and understood the "Tokamak System (TokSys) User Guide" by Walker et al. Finally, an understanding of the underlying physical principals employed by the `gsdesign.m` script is also very important. An understanding of the physical forces acting on the plasma will help in fine-tuning the equilibrium results. A brief review of the Grad-Shafronov derivation is given in Appendix A for the reader if needed.

2.4 Starting File(s) and information

This chapter assumes the reader is already familiar with how to connect to the PCS gate and one of the four servers. If this is not the case, then please refer to Appendix B for detailed instructions on how this should be done.

2.4.1 File(s)

Only one file outside of the TOKSYS library is needed, a gfile from EFIT. However, several inputs to GSDesign need to be loaded into the *Workspace* in `Matlab`. Listed below are the structures, paths, and variables that need to be in the *Workspace* for GSDesign. How to load them will be covered in detail in the next sections. As for the gfile, the reader should refer to Appendix B. Note

Table 2.2: Variables and values needed in the *Workspace* for GSDesign

Name	Value
east_obj_filename	'east_obj_170921.mat'
gatools_root	'/project/builds/TOKSYS/2018-05-18_18-03-12_build138'
GATOOLS_ROOT	'/project/builds/TOKSYS/2018-05-18_18-03-12_build138'
gfile	*1x1 struct*
tok_data_struct	*1x1 struct*

that Appendix B uses a different gfile as an example than this chapter, but the format is the same.

After becoming familiar with the general format of the gfile, move on to a closer look at the gfile used here, g170921.00010, which was provided by Luo et al.

Listing 2.1: Examining the gfile

```
1  EFITD    08/02/2006    #170921    10ms           3 129 129
```

This line provides some valuable information.

- Year: This is important because the EAST objects configuration needs to correspond to 2006 (or whichever year the gfile comes from).
- Grid Size: This comes from the last two numbers in the line. We won't need to use this right away, but later a discussion on how this affects the computation speed and accuracy will be addressed.

Before using this gfile, a few more lines should be explained first.

Listing 2.2: Examining the gfile

```
3562   129   129   170921       10
3563  0.120000000E+01  0.260000000E+01-0.120000000E+01  0.120000000E+01
3564 -0.788019957E+06-0.403353115E+06  0.422465404E+06  0.552643917E+06-0.716103482E+05
3565 -0.206865166E+06  0.196482003E+06-0.300605311E+06-0.965905878E+06  0.351601871E+06
3566  0.194276139E+06-0.261054925E+06
```

Here again is the grid size, shot number, and duration in milliseconds in line 3562, so no new information. However, skipping down to lines 3564 to 3566 to find a total of 12 values. These are the poloidal field coil currents. The reason these are of special note is that the total number of coils EFIT assumed will be crucial for the initial equilibrium. While the coils at EAST are fairly permanent, how they are connected can change the effective number of coils. More precisely, this gfile only gives information about the coil circuits as opposed to the coils themselves. How this is applied to the `gsdesign.m` script will be discussed later. Finally, note that the line numbers are specific to this gfile. Other gfiles will have the same information but at different line numbers. To

find this information, it is easiest to search for the grid size, in this case, 129 129 which is displayed in the first line and the lines preceding the poloidal field coil currents.

2.4.2 TOKSYS

After examining the gfile it is convenient to establish a `.bashrc` script. This script is run from the Linux `[Terminal]` window before `Matlab` is run and sets the global variable GATOOLS_ROOT to a specific path. For `gsdesign.m` to work as described in this chapter, this variable needs to be set to the build from 2018-03-02. As updates are made to the TOKSYS there is no guarantee that the steps detailed here will work for later builds of TOKSYS. It is recommended that the reader completes these steps before using a later build of TOKSYS. For this look at the following bash script named `gsdesign.bashcr`.

Listing 2.3: `gsdesign.bashrc` script to be executed from the Linux `Terminal`

```
1 # gsdesign.bashrc
2
3 # Set paths
4 export MATLABPATH=/project/builds/TOKSYS/2018-03-02_16-31-54_
     build124/startups:$MATLABPATH
5 export GATOOLS_ROOT=/project/builds/TOKSYS/2018-03-02_16-31-54_
     build124
6 export PATH=/project/builds/anaconda2/bin:$PATH
7
8 # User specific aliases and functions
9 alias ls="ls -alh --color"
10 alias matlab="matlab2016a -softwareopengl"
```

Line 1 is the name of the file, lines 4 and 5 set some additional paths that are not necessary but are good to include if scripts other than `gsdesign.m` will be used. In lines 4 and 5, there are two different directory paths, the path in line 6 is all that is needed for now. Here the # symbol comments out line 7 which uses the most recent build of TOKSYS. Then in lines 10 and 11 aliases

are set to make executing the `ls` command give a preferred output and to execute a specific version of `Matlab`. While line 10 is just a personal preference, line 10 sets the command `Matlab` to open `Matlab2016a` and set what graphics rendering package to use. The steps that follow might not work smoothly for all versions of `Matlab`, so it is best to use the version specified here.

A final note is made about where these files should be stored. Each user at EAST is granted a directory named with his or her user name. It is up to the user to then create subdirectories as he or she needs. A common suggestion is that a separate subdirectory is made labeled EAST and used as the working location where all of the EAST related work is kept. The bash script that is shown above automatically changes to this EAST directory and so the bash script itself can be saved directly in the user's directory. Lastly, this EAST subdirectory is also the location from which `Matlab` should be opened to match the steps and procedures that follow.

2.5 Running the scripts

Begin by opening a terminal in the location of the EAST subdirectory. Execute the following lines.

Listing 2.4: Lines and output executed directly from the `Terminal`

```
1 [daw@node60 ~]$ cd EAST
2 [daw@node60 EAST]$ source .bashrc
3 [daw@node60 EAST]$ ssh -X cs1
4 daw@cs1's password:
5 Last login: Wed May 16 15:55:35 2018 from 202.127.205.60
6
7
8 ************************************************************
9
10  Welcome to EAST Computing Server 1 (CentOS6.7-64bit)
11
```

 Negative-Triangularity Configuration on EAST

```
12  Please contact wangfeng@ipp.ac.cn if you have any issues
13
14 *************************************************************
15
16 [daw@cs1 ~]$ matlab
17 MATLAB is selecting SOFTWARE OPENGL rendering.
```

Line 1 changes the directory, line 2 runs the `gsdesign.bashrc` script, and then line 3 opens a secure shell to server 1, followed by entering the user's EAST password on line 4. Lines 6-15 are the output after a successful login. Note that as the user types in the password, the cursor within the terminal does not move. Finally, on line 16 `Matlab` is executed, followed by the output on line 17. At this point the `Matlab` GUI should open. However, it should be noted that some of the quick key commands such as `ctr+c` and `ctr+p` do not behave the same way when using `Matlab` on the server as they do on a PC. For a remedy to this, please see Appendix E.

2.5.1 General start-up

It is convenient to create a start-up script that will add some directory paths needed as well as create some *Workspace* variables.

Listing 2.5: `start_daw.m` start-up script

```
%  Used to set file paths, run toksys_startup, and east_startup
%
%  WRITTEN BY:  David Weldon  ON        2017/10/11
%
%  MODIFICATION HISTORY:
%       2018/04/04  Cleaned up the code to make it more readable
    for a tutorial
%
%
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
9
10 close all
11 clear all
12 clc
13
14 % This is the location of the main directory for all toksys stuffs
15 GATOOLS_ROOT = '/project/builds/TOKSYS/2018-03-02_16-31-54_build124
       ';
16 % GATOOLS_ROOT='/project/builds/TOKSYS/current';
17 gatools_root = GATOOLS_ROOT;
18 addpath(genpath(GATOOLS_ROOT));
19
20 % Get the basics started up
21 % startup
22 toksys_startup
23 east_startup
24
25 % This is the location of all my files
26 local_path = '/home/ASIPP/daw/EAST';
27 addpath(genpath(local_path));
28
29 % Matlab toolbox, needed for running make_east_objects2.m
30 mpc_path = '/pkg/MATLAB/R2016a/toolbox/mpc/mpc';
31 addpath(genpath(mpc_path));
```

Lines 10-12 are optional, when executed in order they: close all figure windows, clear the *Workspace*, and clear the *Command Window*. Lines 15 and 17 are the same as seen in the `gsdesign.bashrc` script and in fact rely on the `gsdesign.bashrc` script for setting the path for the General Atomics Root directory at EAST (GATOOLS_ROOT). The path is then added to the list of paths that `Matlab` uses. Lines 22 and 23 use the default TOKSYS and EAST startup scripts, respectively. Again, this adds paths and sets some *Workspace* variables. Then lines 26 and 27 set

and add the user's directory path, again here `daw` should be replaced with the reader's user name. Finally, in lines 30 and 31, the `Matlab` toolbox is added. For more information on the `Matlab` toolbox, refer to the Tokamak System (TokSys) User Guide.

To create and run this script, go to the `Matlab` GUI and in the top right corner select *New script* as seen below in Figure 2.3. Copy and paste the code from Listing 2.5. Be sure to check for proper

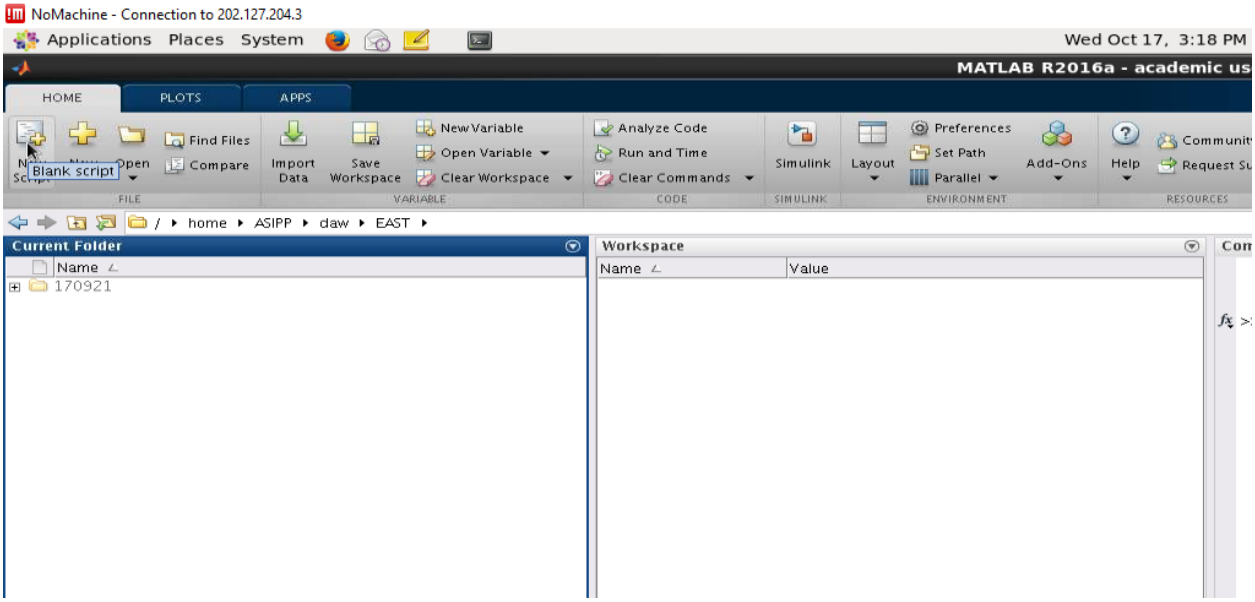


Figure 2.3: Creating a new `Matlab` script

formatting as copy and pasting from this PDF file might not work as expected. Once it has been copied, then save it as `start_XXX` where *XXX* is the user's preference. Here, *daw* is used as this is also the name of the personal directory in this example. Follow this same procedure whenever recreating one of the scripts in this document.

All should now be ready to click on the large green arrow near the top center of the `Matlab` GUI to execute the script. Alternatively, the `F5` key on the keyboard as shown in Figure 4.1 will produce the same result.

2.5.2 GSDesign start-up

Now that the general directory paths and *Workspace* variables are set, a few specific variables are needed for GSDesign to work efficiently. First, establish a directory for all of the outputs. To

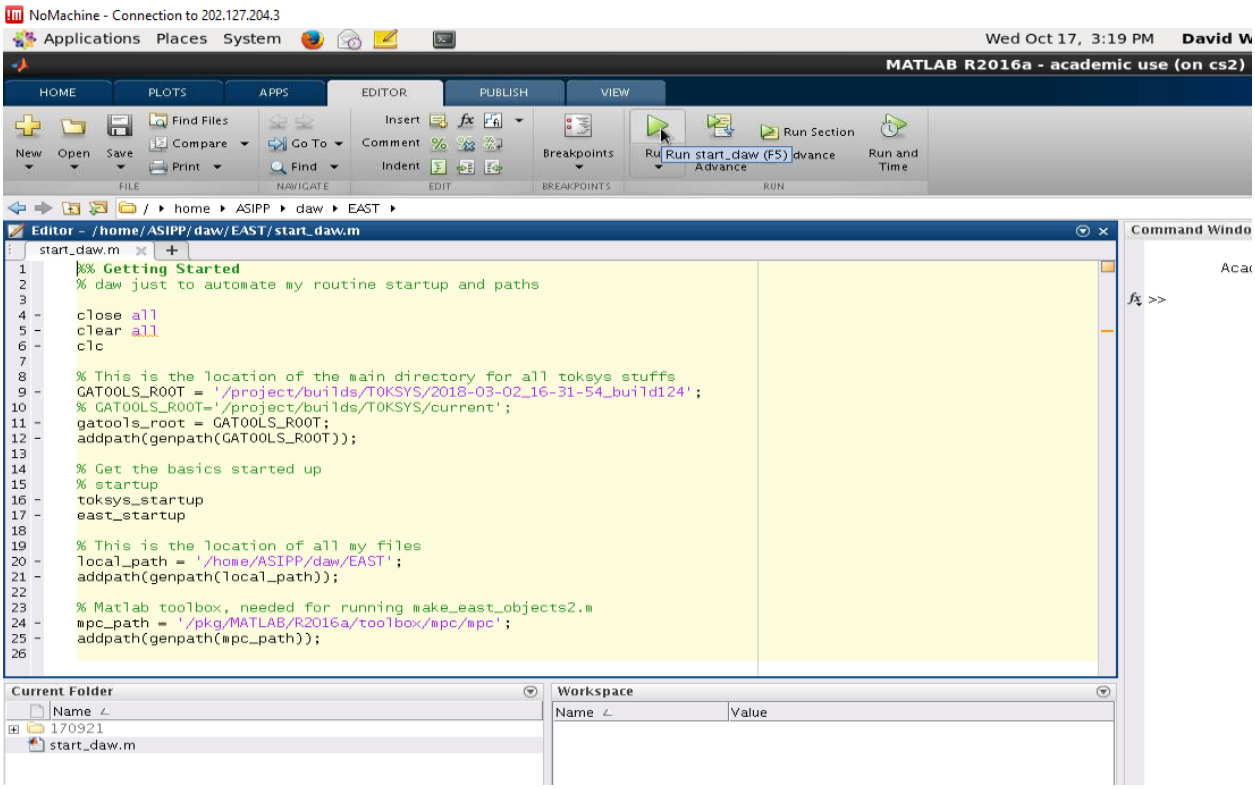


Figure 2.4: Running a `Matlab` script

do this, created a separate directory named after the shot number of this specific gfile, as seen in Figure 2.5. This is also where the original `g170921.00010` file is kept.

This method of organizing the output that will be generated will be very helpful when refining the Grad-Schafranov equilibria because the process could take a dozen attempts or more. In this case, over 17 attempts were needed before a suitable equilibrium was found. With that, now examine the start-up file for GSDesign.

Listing 2.6: `start_gsdesign_daw.m` setting directories, saving inputs/outputs, and additional plots

```
1 %
2 %  Set various file names, save paths, and load variables needed by
3 %  gsdesign to the Workspace. This is also used to build the east
      objects
4 %  needed by gsdesign to match the gfile.
```

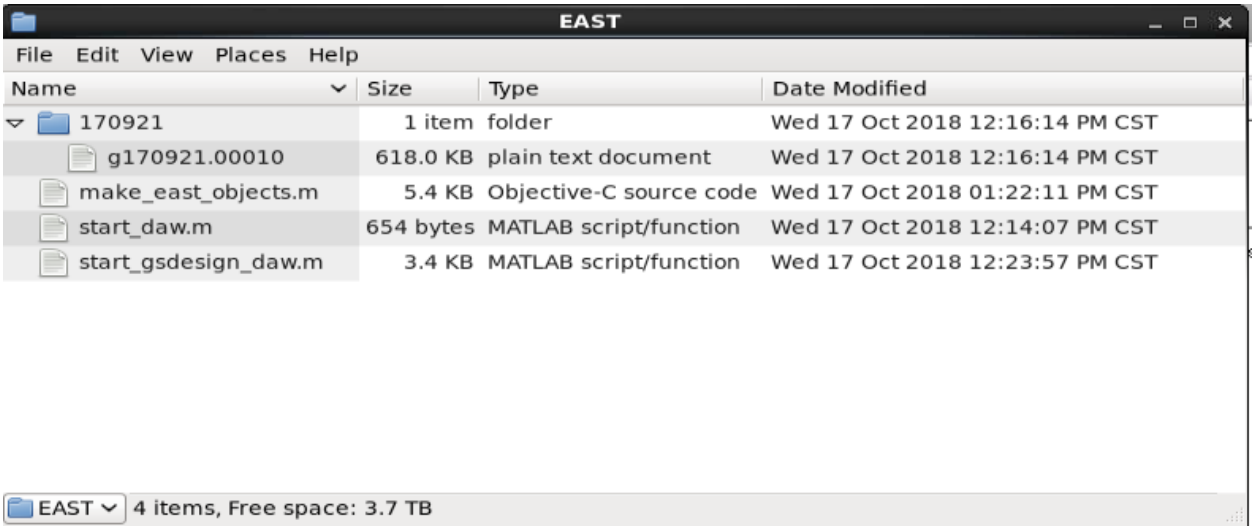


Figure 2.5: Directories and files contained within the EAST directory. The full file path is: /home/ASIPP/daw/EAST

```
5  %
6  %  WRITTEN BY:  David Weldon  ON        2017/10/11
7  %
8  %  MODIFICATION HISTORY:
9  %       2018/04/04  Cleaned up the code to make it more readable
       for a tutorial
10 %
11 %
       %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
12
13 close all % make the GUI nice and clean by closing any previous
       plots
14 clc % make the command window clean as well
15
16 % Choose an EFIT g-file. If you are often working with different
       gfile,
17 % be sure to comment out the ones you are not using.
18 efit_gfile='g170921.00010';
```

```
% efit_gfile='g072659.003500';

% Set the path to where everything should be saved by parsing the gfile
% name and the local path
save_path=cat(2,local_path,'/',efit_gfile(2:7));

disp('Run the demo, this could take a while on 129 X 129 grids') % mostly to let me know the script has run this far

% CHANGE THIS WHEN YOU CHANGE GFILES !!%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% e-coils 12 coils for g170921 but 13 for g072659
gfile = read_gfile_func(efit_gfile,12,1);

% Make the east objects file name
east_obj_filename=cat(2,'east_obj_',efit_gfile(2:7),'.mat');

% Make east objects if needed, but this should only need to be done once
% per gfile. After that, the saved .mat file should be loaded.
% Need to correct the grid size and starting location to match this gfile,
% so I made changes in make_east_objects and saved it in my EAST directory
if exist(east_obj_filename,'file')
    % Load up the objects
    load(east_obj_filename);
else
    make_east_objects
    % to avoid modifying make_tok_objects, just to load the objects created
```

```
44      % by make_tok_objects, save the new .mat where I want, then
            delete the
45      % .mat created by make_tok_objects.
46      load('east_obj_2006_129129.mat');
47      save(cat(2,save_path,'/',east_obj_filename), 'tok_data_struct')
            ;
48      delete('east_obj_2006_129129.mat');
49  end
50  % input('Press ''Enter'' to continue...0','s');
51
52  % Run the specific gsdesign script CHANGE THE V NUMBER!!
53  gsdesign_script=cat(2,'gsdesign_',efit_gfile(1:7),'_',efit_gfile(9:
        end),'v0');
54  eval(gsdesign_script);
55
56  % Save the input that made the equilibrium as well as the results
        from gsdesign
57  input_filename = cat(2,save_path,'/',gsdesign_script,'_input');
58  eq_filename = cat(2,save_path,'/',gsdesign_script,'_eq');
59  save(input_filename, 'spec','init','config');
60  save(eq_filename, 'eq');
61
62  % Save figure by saving the .fig file and by printing it to a pdf
63  fig_filename = cat(2,save_path,'/',gsdesign_script);
64  fig(1) = figure(1); fig(1).PaperType='b4'; fig(1).PaperOrientation=
        'landscape';
65  print(fig(1),fig_filename,'-dpdf','-r0','-fillpage')
66  savefig(fig(1),fig_filename)
```

Lines 13 and 14 have already been discussed in the general start-up so instead focus on lines 18 and 23. Line 18 sets the variable `efit_gfile` to a character string. This string, of course, is the same as the gfile of interest above. If the user is working with more than one gfile, then line 19 might be useful. However, the way these scripts are structured, only line 18 OR line 19 can be

used. If they are both left uncommented then line 19 will just overwrite the variable `efit_gfile`.

Notice in line 23 how the `efit_gfile` character string is parsed to set the `save_path`. As seen below, these two variables will be parsed many times to make various output file names.

Line 25 is just to have some *Command Window* output in `Matlab` to indicate that the script is running.

Lines 27-29 are fairly self-explanatory but are necessary so that the user does not waste several hours of computation while running the wrong number of coils.

Line 32 makes the filename for saving the EAST objects that are created in line 42 and saved in line 47. `tok_data_struct` is the structure containing all the EAST objects and must first be loaded into the *Workspace* to run GSDesign. However, running `make_east_objects` at each start-up is highly inefficient. It is better to run it once and save the structure then load this structure to the *Workspace* for every subsequent start-up, which is done with the `if` statement in line 38 and the `load` command in line 40. The first time this script is run, no `east_obj_filename` will be found and so lines 42-48 will create it and save it. However, the `.mat` file is created using the default settings. To get the EAST objects that are needed for this example, see Appendix D.

Lines 53 and 54 also take previously made character strings to parse and concatenate them with a few other things to call up a script that contains all of the GSDesign settings and the actual call to `gsdesign.m`. This separate script dealing with the settings for GSDesign will be dealt with in Section 2.6.1.

Line 50 is special and should be noted carefully. `Matlab` has a debugging function and it works just fine for some uses. But sometimes a user might want a little more control over where and how `Matlab` stops. So, line 38 or something like it can be used. If the user is exploring a script from TOKSYS that is necessary for GSDesign then one could use several of these lines inserted in the script with different numbers such that one knows which part of the script is being executed.

Lines 57-66 are all about saving the results. The reader is encouraged to organize his or her file directories and results in a way that is most logical to him or her. Take note of lines 65 and 66 as these are saving the same figure but in different file formats.

2.6 Creating an equilibrium

As mentioned above concerning line 57, a character string is made referring to a specific script that has settings for GSDesign to use in a single equilibrium design. The result is the character string `gsdesign_g170921_00010v0.m`. Next is an examination of this script to see how it works.

2.6.1 Matching the gfile equilibrium

The purpose of this version 0 script (hence the `v0` at the end of the file name) is to mark it as the first attempt at converging on an equilibrium and is designed to try and reproduce the exact same shape and coil currents that the gfile has.

Listing 2.7: `gsdesign_g170921_00010v0.m` setting the specifications, targets, weights, initial equilibrium, and configuration for GSDesign

```
%
%  USAGE:   gsdesign_g170921_00010
%
%  PURPOSE: DEMO of gsdesign showing design of EAST single-x-point
%
%  INPUTS:  none
%
%  OUTPUTS: eq, a single-x-point equilibrium
%
%
   %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%
%  WRITTEN BY:  Anders Welander  ON     3/12/14
%
%  MODIFICATION HISTORY: Augmented from gsdesign_demo_d3d_DN by
   David
```

```
15 %  Weldon on 2017-10-10
16 %
17 %  This is meant to get an equilibrium that matches the file
       g170921
18 %
19 %
       %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
20
21 % Load EAST tokamak configuration
22 if exist('east_obj_filename','var')
23   try
24     clear tok_data_struct
25     load(east_obj_filename);
26 %     tok_data_struct.rg; % I don't think this is needed
27   catch
28     error('east_obj_filename does not hold name of a matfile
           containing tok_data_struct')
29   end
30 else
31   east_obj_filename = [getenv('GATOOLS_ROOT'), ...
32     '/tokamaks/east/make/east_obj_2006_129129.mat'];
33   if exist(east_obj_filename,'file')
34     load(east_obj_filename)
35   else
36     east_obj_filename = ...
37       '/m/GAtools/tokamaks/east/make/east_obj_2006_129129.mat';
38     if exist(east_obj_filename,'file')
39       load(east_obj_filename)
40     else
41       disp('Could not find east objects. Please set the variable:')
42       disp('east_obj_filename')
```

```
      disp('to name of matfile containing tok_data_struct for east')
    end
  end
end

config = tok_data_struct;
config.constraints = 1;
config.psikn = [0 0.40 0.70 1];
config.no_edge_current = true;
config.no_edge_gradient = true;
config.plot_settings.SOL.n = 9;
config.plot_settings.SOL.d = 1e-3;
init = [];

clear spec gsdesign

% Specify points rsep, zsep where flux should equal the boundary flux
spec.targets.rsep = gfile.rbbbs;
spec.targets.zsep = gfile.zbbbs;

% Specify points rx, zx where the poloidal field should vanish
% Just because there is only 1 null, doesn't mean having 2 x-points isn't a
% good idea. Also, read the documentation to see how to put a box where no
% x-point should appear.
[~, ix1] = max(spec.targets.zsep);
spec.targets.rx = spec.targets.rsep(ix1);
spec.targets.zx = spec.targets.zsep(ix1);
spec.weights.x = 1;
```

```
71
72 % The SN can be more balanced with higher weights on the x-points
73 spec.weights.sep = 10*ones(1,length(spec.targets.rsep));
74 spec.weights.sep(end-1:end) = 50*ones(1,2);
75 spec.weights.sep(ix1) = 10;
76
77 spec.targets.cpasma = gfile.cpasma;
78 spec.weights.cpasma = 1;
79
80 spec.cccirc = [config.def_connect.fcid 13 -13];
81 config.cccirc = spec.cccirc;
82
83 i = length(spec.cccirc) - length(gfile.brsp);
84 j = zeros(i,1);
85 ci = [gfile.brsp;j];
86 fcnturn = config.fcnturn;
87 fcnturn(7:10) = fcnturn(7)+fcnturn(10);
88
89 spec.locks.ic = (sign(spec.cccirc)'.*ci(abs(spec.cccirc)))./fcnturn
    ;
90 spec.locks.ic(15:16) = zeros(2,1);
91
92 clear i j ci1; % keeps the workspace clear of intermittent clutter
93
94 % Call gsdesign with these specs
95 % In this first design, it is better to not include the prims right
     away,
96 % otherwise gsdesign can't get to the spec.lock.ic values.
97 config33 = regrid(33,33,config);
98 eq33 = gsdesign(spec, init, config33);
99 disp('The 33x33 grid equilibrium without ffprim or pprime has been
    calculated');
```

```
100 % Now we include the p' and ff' and use the previous equilibrium as
        the
101 % init
102 disp('Next, the 33x33 grid equilibrium will be calculated including
        the prims')
103 config.pprime0 = gfile.pprime;
104 config.ffprim0 = gfile.ffprim;
105 config33_prims = regrid(33,33,config);
106 eq33_prims = gsdesign(spec, eq33, config33_prims);
107 disp('The 33x33 grid equilibrium with ffprim or pprime has been
       calculated');
108 % Now we change the grid size back to 129x129 and again use the
       previous
109 % equlibrium as the init
110 disp('Next, the 129x129 grid equilibrium will be calculated
       including the prims')
111 eq = gsdesign(spec, eq33_prims, config);
```

The header of the script in lines 1-19 explains a few details about the script. First of all that it was originally a demo file that comes with TOKSYS located in the same directory as GSDesign. Note that while on line 6 it says there are no inputs, that is not entirely true. When calling the script it is true that no input *arguments* are needed, however the *Workspace* must contain specific structures, data, and paths as listed under Section 2.4.1 in table Table 2.2.

Before going further, it would be enormously helpful for the reader to look at Appendix F which is the help file for GSdesign included in TOKSYS. For more convenient reference, this file can also be accessed by going to the *Command Window* in `Matlab` and typing in `help gsdesign`.

Lines 21 to 46 are taken directly from `gsdesign_demo_d3d_DN` and have not been modified. This is mostly to check and find an appropriate objects file for EAST. This is taken care of by line 25. If the user is using a different EAST objects file, then it should be changed here. But, if the EAST objects that was created by from `make_east_objects` has already been loaded, then the *Workspace* should already contain the structure `tok_data_struct`.

Next, in lines 48-55 several values and flags are being set for the `config` structure. Almost all of these are the defaults that came from the original DEMO file except for line 48. There are several key pieces of information in the `tok_data_struct` that are also needed in the `config` structure.

Line 57 is also part of the original DEMO file and is just to ensure that if there is a `spec` or `gsdesign` in the *Workspace*, it is completely cleared before the subsequent ones are created.

Lines 60-75 are pretty self-explanatory. The targets come from the gfile while the weights are up to the user to decide. There is not an exact way to determine how much to weight the boundary and x-points. This is a trial-and-error process to determine it, but with experience comes an intuitive understanding of how to set them.

Lines 77 and 78 set the plasma current to the value given in the gfile. The weight of which was determined by the trial-and-error method starting with:

```
spec.weights.cpasma = 0.001
```

and gradually increased until it did not have the largest error at the end of the fitting process.

Lines 80 and 81 set the circuit configuration of the coils. This is not difficult, but needs to be addressed. Typing "`tok_data_struc.def_connect.fcid`" into the *Command Window* will give the output:

```
» tok_data_struc.def_connect.fcid

ans =

                                        1 7 2 8 3 9 4 10 4 10 5 11 6 12
```

This vector is essentially showing how the coils are to be connected to each other. For example, coil 1 is connected to itself because it is in the first position. Then coil two is actually coil 7. Why? That is because of a difference between the order of coils that the gfile assumes and that which GSdesign uses. This re-ordering is done for all the coils. It should also be noted that coils in positions 7 and 9 are connected together in series as are those in positions 8 and 10. Finally, in Line 80 `13 -13` is added because the final two coils are connected together in an anti-series configuration.

However, this is just part of the battle with the differences between the gfile and GSdesign. Next, in lines 83-90, it is necessary to perform a calculation to convert the coil currents of line 90 and divide them by the number of turns in each coil (after the ".* in line 89) while still maintaining the correct sign for each coil current (before the ".*). The result of which is the current per turn of each coil (here it is meant each circuit because some coils are connected) as the `spec.lock.ic` value. In line 90 the final two coils are locked to zero because they are not used in designing the equilibrium but are used in the feedback control of the plasma.

Finally, all is ready to actually run GSDesign and get an equilibrium. To do this as efficiently as possible, line 97 first "re-grids" the equilibrium to 33 by 33 mesh points. This requires much less computation and will work as a first "guess" so to speak. Without this re-grid step, the first iteration can take several minutes. If there are any mistakes in the settings or elsewhere, they will most likely be seen in the first iteration, so this is a way to save time. Line 98 is the first run of GSDesign and gives the result `eq33` meaning the equilibrium in a 33 by 33 grid. The inputs, which are better explained in Appendix F, are the specifications set above, the initial equilibrium, `init` is what GSDesign uses as a starting point. For line 98, `init` is empty as set by line 55. Lastly, the 33 by 33 configuration is included in line 98. Lines 99 and 102 are notes that show up in the *Command Window* to indicate which step in the computation is being executed.

Lines 103-105 include the p' and ff' values from the gfile in the configuration so that an even better fit can be found. Then GSDesign is run again, but the previous `eq33` is used as the new `init`. The resulting equilibrium is called `eq33_prims`, and two messages are displayed as before.

For the final calculation, the `eq33_prims` equilibrium is used as the new `init` and a 129 by 129 grid configuration with p' and ff' as the `config` in line 111 resulting in the final equilibrium, `eq`.

2.6.2 GSDesign results

With the run file, `gsdesign_g170921_00010v0.m` finally explained, an examination of the outputs is in order. Figure 2.6 is the main figure to look at. This is the default output from GSDesign. It is

not displayed here, but in the figure window the "?" can be seen in the top row. Clicking this will bring up Figure 2.7, which gives a fairly detailed explanation for everything that is in Figure 2.6.

While this information is quite useful, a little more needs to be said about the error plots. In the lower-left corner is a subplot showing all the errors of all the target values. Zooming in on this plot will show the labels of each target however, for this example, the coil currents will not be found because they have been locked to the exact values from the gfile. Only targets such as boundary flux points and the plasma current will be found in this subplot. In fact, for this equilibrium, the largest positive error is a separatrix boundary point, highlighted in red and the largest negative error is another separatrix point, highlighted in blue. On any of these subplots, a double-click on any area will zoom in, or clicking and dragging to make a rectangular zoom area will zoom in as well. Finally, right-clicking will reveal some options including returning to the normal zoom. This can easily be done once GSDesign has completed the calculation. Attempting to zoom in or out while GSDesign is running will probably result in a lot of lag. Moving on to the immediate right of the above-mentioned subplot is another subplot of errors and this is a bit more important. Note here that all the error vectors are nice smooth curves and the on the y-axis is 10^{-11}. This is showing how good the convergence is. For matching the gfile, this is as good as can be achieved because so many values are locked to match the gfile exactly. While 10^{-11} does indeed seem like a pretty small error, it is in fact not that good. The desired error would be on the order of 10^{-14} or even 10^{-15} because this means the error in the convergence would be almost nothing and the error vectors would be just noise. Ideally, the only error should be noise that is inherent in any numerical calculation arising from floating-point precision. So, for this case, there is still some fundamental error in matching the gfile exactly. Where that is arising from, is not clear. However, the main point of matching the gfile is an exercise in using GSDesign correctly. So while there is still some error, if the user feels confident in the operation of GSDesign, then move on to modifying the shape in Section 2.7.

Moving on to the newly created files as shown in Figure 2.8. First is the `east_obj_170921.mat` which was created by line 47 of Listing 2.6, next is the gfile, then the `.fig` file that is the result of line 66 of Listing 2.6, followed by the `.m` file which has already been discussed in Listing 2.7, then

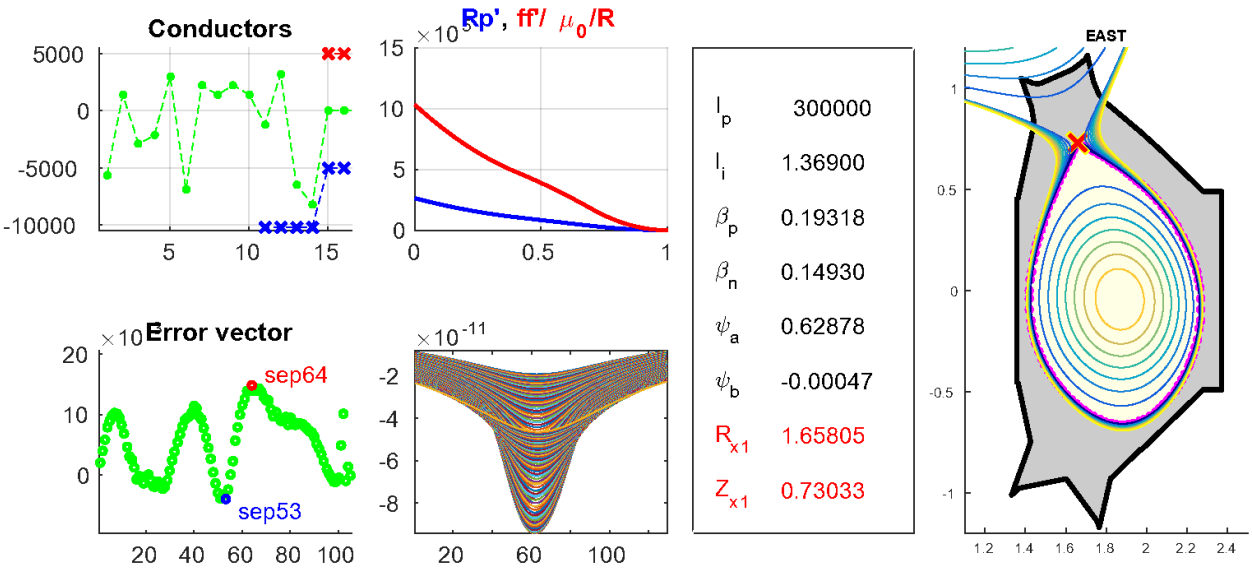


Figure 2.6: Results from matching the gfile

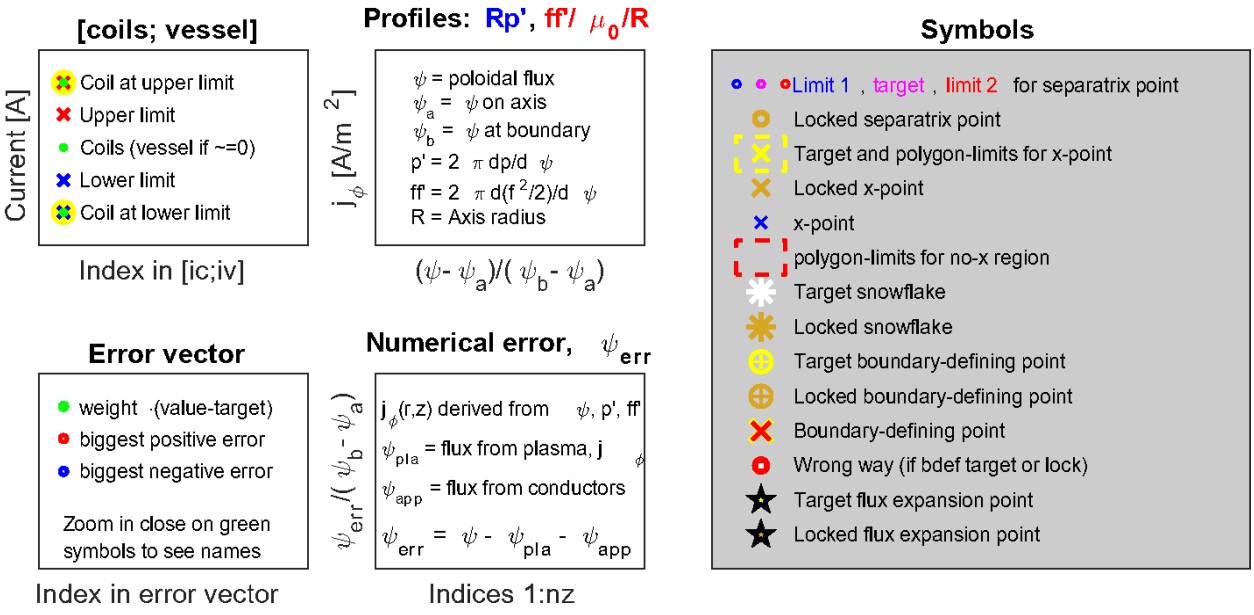


Figure 2.7: The explanation of the GSDesign results figure

the `.pdf` file was created by lines 64 and 65 in Listing 2.6, and finally the last two `.mat` files are as their names claim, the equilibrium and the inputs. These two files are created by lines 57-60 in Listing 2.6.

`gsdesign_g170921_00010v0_input.mat` is simply a single file containing the structures `spec`, `init`, and `config` which were all in the `gsdesign_g170921_00010v0.m` file.

`gsdesign_g170921_00010v0_eq.mat` is the equilibrium that gsdesign outputs as the *Workspace* variable `eq` and is a structure. Going to the *Command Window* type in "`eq`" to see what is contained in this equilibrium. To see what each of these fields is, try typing in "`eq.descriptions`".

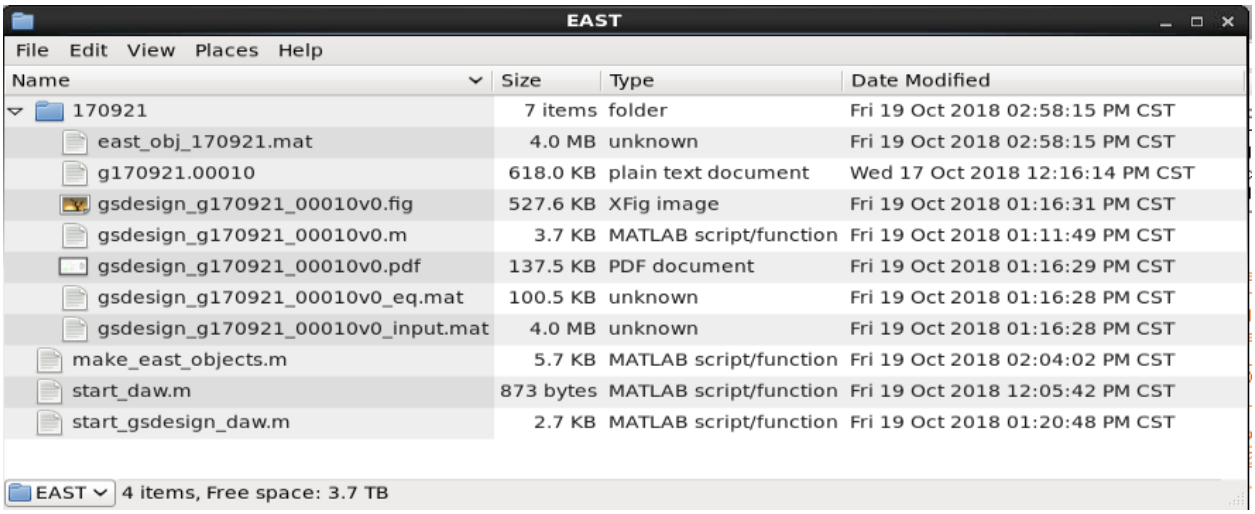


Figure 2.8: EAST directory containing all the new files that should have been created by running GSDesign as well as the files previously shown in Figure 2.5

It is now up to the user to explore the contents of the `eq` structure depending on what the user needs from it. In the case of using `gsdesign.m` to design an equilibrium, then the coil currents, fluxes, betas, inductance, and other such parameters are probably most important.

2.7 Designing a negative-triangularity equilibrium

This section follows immediately from the previous section and details several more steps in the design process, namely: slightly rotating the plasma boundary, shifting the boundary vertically, expanding/contracting the shape, adding in divertor target points, and finally adjusting

the coil currents to be within safe operating conditions. This is all done in a new file called `gsdesign_g170921_00010v1.m` which will use `gsdesign_g170921_00010v0_eq.mat` as the `init` as shown in Listing 2.8, line 70.

Listing 2.8: `gsdesign_g170921_00010v1.m` is created by changing specific parts of `gsdesign_g170921_00010v0.m`. Note, the line numbers do not match exactly with Listing 2.7 because more was added to the preamble notes in lines 1-30

```
66 config.plot_settings.SOL.d = 1e-3;
67
68 % Get the previous equilibrium and use it as the initial
       equilibrium
69 load('gsdesign_g170921_00010v1_eq.mat')
70 init = eq;
71
72 clear spec gsdesign eq
73
74 % Specify points rsep, zsep where flux should equal the boundary
       flux
75 rsep = gfile.rbbbs;
76 zsep = gfile.zbbbs;
77 R0 = gfile.rmaxis;
78 Z0 = gfile.zmaxis;
79
80 % Here I want to manipulate the shape a bit by bringing in the
       outermost
81 % flux surface a little bit but keeping the other points the same
82 [~, ix1] = min(zsep);
83 [~, ix2] = max(zsep);
84 zmin = zsep(ix1);
85 rmin = rsep(ix1);
86 zmax = zsep(ix2);
```

```
87 rmax = rsep(ix2);
88 rline = (zsep-zmax)*(rmax-rmin)/(zmax-zmin) + rmax;
89 K = 0.95; %right side R reduction
90 rsep(min(ix1,ix2):max(ix1,ix2)) = K*(rsep(min(ix1,ix2):max(ix1,ix2)
       ) ...
91     -rline(min(ix1,ix2):max(ix1,ix2)))+rline(min(ix1,ix2):max(ix1,
           ix2));
92 spec.targets.rsep = rsep;
93 spec.targets.zsep = zsep;
94
95 % Here I want to bring the x-points closer by C%, and to keep the
       shape the
96 % same, I need to do the same to all other points along the
       boundary. The
97 % horizontal centerline is already calculated as Z=0, but now I
       need to
98 % know what R is in the center of the vessel. I also want to apply a
99 % rotation to the shape by t radians.
100 C = 1.05;
101 Rshift = 0.0;
102 Zshift = 0.08;
103 t = 1*pi/200;
104 rot = [cos(t),-sin(t);sin(t),cos(t)];
105 sep0d = rot*[spec.targets.rsep-R0 spec.targets.zsep-Z0]'*C;
106 spec.targets.rsep = sep0d(1,:)' + R0 + Rshift;
107 spec.targets.zsep = sep0d(2,:)' + Z0 + Zshift;
108 clear C R0 Z0 Rshift Zshift rline zmin rmin zmax rmax rsep t rot K
109
110 % Specify points rx, zx where the poloidal field should vanish
111 % Just because there is only 1 null, doesn't mean having 2 x-points
       isn't a
112 % good idea. Also, read the documentation to see how to put a box
```

```
    where no
% x-point should appear.
[~, ix1] = max(spec.targets.zsep);
spec.targets.rx = spec.targets.rsep(ix1);
spec.targets.zx = spec.targets.zsep(ix1);
spec.weights.x = 1;

% Add a few boundary points where we want the strike points of the
    legs
rsp = [1.43 1.72]';
zsp = [0.92 1.06]';
spec.targets.rsep = [spec.targets.rsep;rsp];
spec.targets.zsep = [spec.targets.zsep;zsp];

% Adjust weights on the boundary points, strike points, and x-
    points,
spec.weights.sep = 10*ones(1,length(spec.targets.rsep));
spec.weights.sep(end-1:end) = 50*ones(1,2);
spec.weights.sep(ix1) = 10;

% Probably only need to specify these three parameters
spec.targets.cpasma = gfile.cpasma;
spec.weights.cpasma = 0.0001;
spec.targets.li = 1.2; %eq.li;
spec.weights.li = 1;
spec.targets.betap = 1.0; %eq.betap;
spec.weights.betap = 1;
% Design the beginning and ending flux levels of a shot
spec.targets.psibry = 2.43;
spec.weights.psibry = 10;

% spec.cccirc = [1 2 3 4 5 6 7 8 7 8 9 10 11 12 13 -13];
```

```
142 spec.cccirc = [config.def_connect.fcid 13 -13];
143 config.cccirc = spec.cccirc;
144
145 spec.targets.ic = zeros(16,1);
146 spec.weights.ic = 0.00002*ones(length(spec.targets.ic),1);
147
148 spec.locks.ic = nan(16,1); % must have this before specifying any
    locks on ic
149 spec.locks.ic(15:16) = zeros(2,1);
150 spec.limits.ic = [12.8*ones(6,1);11.6*ones(4,1);10.2*ones(4,1)
    ;5;5]*[-1000 1000]*0.89;
151
152 % Call gsdesign with these specs
153 disp('The 33x33 grid equilibrium with ffprim or pprime will be
    calculated');
154 config.pprime0 = gfile.pprime;
155 config.ffprim0 = gfile.ffprim;
156 config33_prims = regrid(33,33,config);
157 eq33_prims = gsdesign(spec, init, config33_prims);
158 eq33_prims = gsdesign(spec, eq33_prims, config33_prims);
159 % eq = eq33_prims;
160 disp('Next, the 129x129 grid equilibrium will be calculated
    including the prims')
161 % input('Press ''Enter'' to continue...','s');
162 disp('I hope it works')
163 eq = gsdesign(spec, eq33_prims, config);
```

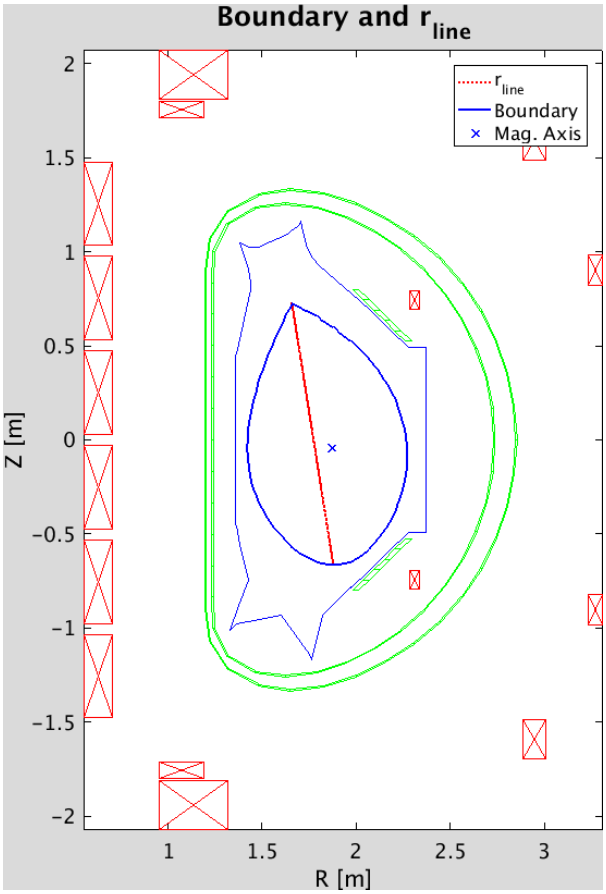
2.7.1 Boundary rotation, shift, and expansion/contraction

The boundary points of the equilibrium made in Section 2.6 can be found by loading the equilibrium into the `Matlab` *Workspace* by either double-clicking the file or by using the command "`load 'gsdesign_g170921_00010v0_eq.mat'`", as in line 69, in the *Command Window* then checking

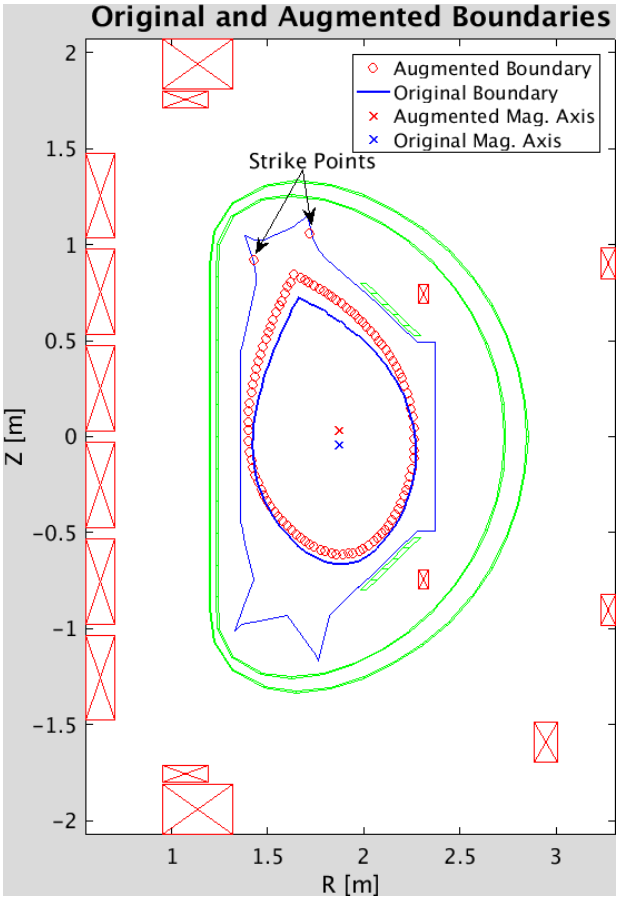
`eq.rbbbs` and `eq.zbbbs`. In Listing 2.8, lines 75-76, these boundary points are still taken directly from the gfile but could come from the previous equilibrium. However, these boundary points are not exactly the same. GSdesign can be set to "lock" the boundary points or set them as "targets." In either case, GSdesign will produce boundary points that are not exactly the same as the input. If GSdesign is set to "lock" them, then there will still be some rounding error, however small; if GSdesign is set to "target" them, then they can vary quite a bit. This is why in Listing 2.8 the boundary points from the gfile are used to eliminate error propagation from one version to the next as this `"v1"` is the first of 17 versions to achieve optimal results.

With that said, the next step is to actually augment the boundary points. Lines 77-78 set the magnetic axis of the plasma as the center about which the shape will be rotated. Lines 82-87 find the highest and lowest boundary or separatrix points (`rsep, zsep`), then line 88 creates r_{line} connecting them as seen in Figure 2.9a. Then line 89 sets `K` as the expansion/contraction factor and lines 90-91 take the horizontal distance of each point to the right of the r_{line} and multiplies it by this expansion/contraction factor, effectively pulling in or pushing out the right side of the boundary. Determining which side of the r_{line} gets augmented is done by knowing that the boundary points are ordered clockwise from the upper x-point which is done at the beginning of line 90 `rsep(min(ix1,ix2):max(ix1,ix2))`. If the `min` and `max` were reversed then the left side of the line would be augmented. Lines 92-93 then overwrite the previous boundary points with the augmented ones. Here the `K` factor is set to 0.95 as this proved to be a good balance between plasma volume and decreasing the lower triangularity. Note, that because this modifies b (vertical minor radius), the elongation κ is also increased, as discussed in Section 1.2.3.

Next, a universal expansion/contraction factor (not just for the left or right side and not just horizontal), a vertical shift, and a rotation to the boundary are added. Lines 95-99 explain this as well, but not in great detail. Line 100 establishes the expansion/contraction factor `C`, increasing the area of the boundary slightly. Lines 101-102 set the horizontal and vertical shift. It was best to keep the plasma horizontally unchanged but move it up slightly. The vertical adjustment was a compromise between decreasing δ_L and having the open field lines strike the divertor target plates



(a) Boundary of gfile g170921 with a line connecting the highest and lowest points, allowing the points left or right of the line to be moved toward or away from the line by a percentage of their original distance from the line.



(b) Original and augmented boundary of gfile g170921 with original and augmented plasma center. The augmented boundary includes a contraction on the right side of 0.95, anti-clockwise rotation of $\pi/200$, overall expansion from the center of 1.05, and a shift upwards of 0.08 m.

Figure 2.9: Comparison between the starting PTC boundary and the final NTC boundary design

safely as will be discussed in Section 2.7.2. Line 103 sets the rotation angle $t = \pi/200$ rad which is a minute anti-clockwise rotation. Then a simple rotation matrix is set up in line 104 and is shown more clearly in Section 2.7.1. Where rot is the rotation matrix and t is the rotation angle.

$$rot = \begin{bmatrix} \cos(t) & -\sin(t) \\ \sin(t) & cos(t) \end{bmatrix} \tag{2.1}$$

And finally, lines 105-107 tie the rotation, expansion, and shift altogether, more easily understood in Section 2.7.1. Where S is the matrix containing the ordered pairs of the augmented separatrix/boundary points, $r_{1\Gamma n}, z_{1\Gamma n}$ are the r and z coordinates of each separatrix/boundary point, R_0 and Z_0 are the coordinates of the magnetic axis and are taken to be the center of the plasma, C is the expansion/contraction factor, and R_{shift} and Z_{shift} are the horizontal and vertical shifts applied to the boundary, respectively.

$$S = \begin{bmatrix} \cos(t) & -\sin(t) \\ \sin(t) & cos(t) \end{bmatrix} \cdot \left(\begin{bmatrix} r_1 & z_1 \\ r_2 & z_2 \\ \vdots & \vdots \\ r_n & z_n \end{bmatrix} - \begin{bmatrix} R_0 & Z_0 \end{bmatrix} \right)^T C + \begin{bmatrix} R_0 + R_{shift} & Z_0 + Z_{shift} \end{bmatrix} \tag{2.2}$$

The results of these boundary augmentations can be seen more clearly in Figure 2.9b. And to close this section, line 108 clears all the intermediate variables that were used in the above calculations but are not used elsewhere.

2.7.2 X-points and strike points

Starting with lines 110-117 are the explanation for and creation of the x-point as seen in Figure 2.6 in the rightmost subplot designated by the red "X" at the top of the separatrix. This can be adjusted, but because it is based on the boundary that was just modified it will already select the best place

for the x-point. However, the user augments the code, keep in mind that line 114 assumes that no strike points have been made and so the highest boundary point should still be on the separatrix. The next set of lines examine the strike points. If the strike points are created before the x-points it could cause the x-point to be placed on a strike point. Lastly, line 117 sets the weight. This will need to be adjusted to balance the shape and where the strike points land.

Next, lines 119-123 set two strike points as seen in Figures Figure 2.9b and Figure 3.1a. The exact location of these strike points was determined by looking at the physical structure of the divertor target plates. The goal was to avoid having the open field lines striking the critical areas of the divertor target plates which are demarcated in Figure 2.10. The weights for the boundary, x-, and strike points are all set in lines 125-128. How much to weight each one is a trial-and-error process; a battle among all the desired characteristics of the plasma.

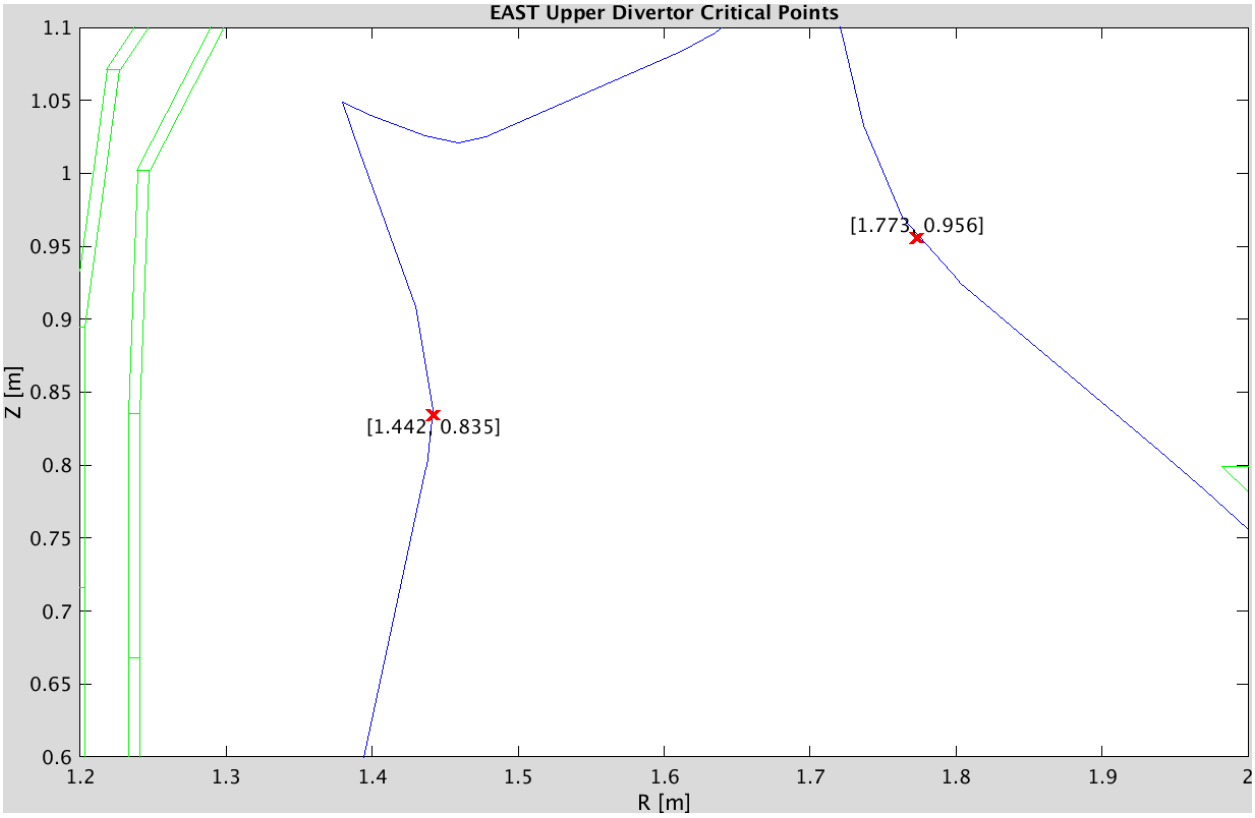


Figure 2.10: Closeup of the critical points in the upper divertor on EAST

2.7.3 Basic plasma parameters

This section discusses the basic plasma parameters that are shown in the subplot of Figure 2.6 second from the right. The key parameters of which are shown in Table 2.3. These are typically the parameters used to characterize the toroidal current density in the plasma.

Table 2.3: Summary of basic plasma parameters that are often targeted in GSdesign

Parameter	Symbol	Units	TOKSYS	Equation
Plasma Current	I_p	[MA]	`cpasma`	$I_p = \int_{\Omega_p} J_\varphi \, \mathrm{d}S$
Internal Inductance	l_i	[H]	`li`	$l_i = \frac{4}{\mu_0 r_c I_p^2} \int_{V_p} \frac{B_p^2}{2\mu_0} \, \mathrm{d}\varphi$
Poloidal beta	β_p		`betap`	$\beta_p = \frac{4}{\mu_0 r_c I_p^2} \int_{V_p} \langle n k_B T \rangle \, \mathrm{d}\varphi$
Boundary Flux	ψ_b	[V-s]	`psibry`	$\psi_b = \psi(r,z)\big\|_{LCFS}$

The Plasma Current is the total flow of electrons and ions through the plasma within the separatrix then integrating the current density in the toroidal direction J_φ over the area designated by the poloidal cross-section within the separatrix Ω_p gives the plasma current I_p. The Internal Inductance is the self-inductance of the current loop that is within the volume of the plasma, created by the plasma current and is calculated by taking the magnetic field that is produced by the plasma current (Ampére's Law $2\pi r_c B_\vartheta = \mu_0 I_p$) and integrating over the volume of the plasma V_p in the toroidal $\partial\varphi$ direction. Here, r_c is the horizontal coordinate of the centroid of the plasma and can be thought of as the current axis and B_ϑ is the poloidal magnetic field created by the plasma current. The beta is a ratio of plasma pressure to magnetic pressure. Take the Poloidal beta β_p which only looks at the magnetic pressure from the poloidal magnetic field B_ϑ. The plasma pressure is calculated with $p = nk_BT$ and averaged over the volume of the plasma. Finally, the Boundary Flux is the flux evaluated at the boundary and is not easily calculated but a simplified solution assuming a circular cross-section is given in Appendix A. This is an important parameter as it is directly related to the flux [V-s] the poloidal field coils can give the plasma. In reality, all the flux in the plasma is effected but the flux at the boundary is what the diagnostics can measure. The relationship between

them can be seen better in Figure 2.11 and is discussed in more detail in Section 2.7.4.

The remainder of file `gsdesign_g170921_00010v1.m` from line 141 onward are fairly straight forward. The major difference between it and Listing 2.7 is that the poloidal field coil currents are no longer locked to the gfile values and are instead allowed to adjust however needed to match the previously mentioned parameters. This is set in line 145 where the targets are set to zero so as to keep the coil current far from their limits. Line 148 also sets the locks to `NaN` to be sure only the targets will influence the convergence. Lastly, line 150 sets the limits of the coil currents. These limits are taken from the database containing the physical limits of each coil and are displayed in Table 2.4 as well as the "soft" limits of each coil set to 89% of the "hard" or physical limits. These hard limits are also shown in Figure 2.1a.

Table 2.4: Hard (physical limit due to mechanical stresses and cooling ability), soft limits (89% of the physical limits), and power supply voltage limits of the poloidal magnetic field coil currents.

Coil Numbers	1-6	7-10	11-12	13-14	15-16
Hard Limit [kA]	±12.8	±11.6	±10.2	±10.2	±5.0
Soft Limit [kA]	±11.4	±10.3	±9.1	±9.1	±4.5
Voltage Limit [V]	±350	±800	±400	±330	±1600

2.7.4 Flux

As mentioned before, there is one last topic to discuss when it comes to designing an equilibrium. As seen in Figure 2.11, the design of an equilibrium is essentially just a single time slice during a discharge "flat-top" as show in the second subplot. This is all well and good, but it doesn't indicate if there will be enough flux available from the coils to reach the designed equilibrium from the end of the ramp-up. The ramp-up is the phase where the plasma current is steadily increased at the beginning of the discharge, directly preceding the flat-top shown in Figure 2.11. For this, one can perform a linear extrapolation such that the maximum voltage the power supplies of the individual poloidal magnetic field coils are not exceeded. These limits are also shown in Figure 2.1b. First, it is necessary to make sure the power supplies can handle the transition from the ramp-up phase to the

shape control phase at flat-top. To do this, calculate the maximum voltage that will be demanded to change the current of each coil as shown in Equation (2.3).

$$M_{cc}\frac{\mathrm{d}I_{PF}}{\mathrm{d}t} = V_{PS} \tag{2.3}$$

Where M_{cc} is the square matrix containing the coil-to-coil mutual inductance, I_{PF} is the vector containing the current in each coil, and V_{PS} is the vector containing the voltage demanded from the power supply. This equation can be discretized as shown in Equation (2.4).

$$M_{cc}\frac{I_{PFf} - I_{PFi}}{t_f - t_i} = V_{PS} \tag{2.4}$$

As long as the voltages in V_{PS} remain within the range designated in Table 2.4 then the equilibrium can be reached from the end of the ramp-up. If not, then the designed equilibrium needs to be adjusted. This is not an exact process, but again, a trial-and-error process that is based on the ramp-up of a similar discharge that has already succeeded. In Figure 2.11, the vertical line marking `GS Design` marks the extrapolation from the equilibrium shown in Figure 3.1a. This is the maximum flux that can be expected in the plasma, based on Equation (2.4). Any higher than this and the coils cannot reliably produce enough flux. This establishes the rough estimate for when to employ the RZIP or ISOFLUX control to shape the plasma and is discussed in the next section.

2.8 PCS parameters

Before closing this chapter, it is necessary to address the Plasma Control System (PCS). The PCS was inherited from DIII-D and modified for the EAST central control system and has been in use since the first plasmas at EAST in 2006 [28]. The modifications to it were necessary because the special control characteristics that are a result of the coils having an integrated design (used to induce the plasma current and control the shape of the plasma) and being superconducting. The NTC follows the normal divertor plasma discharge control sequence using a piece-wise series of

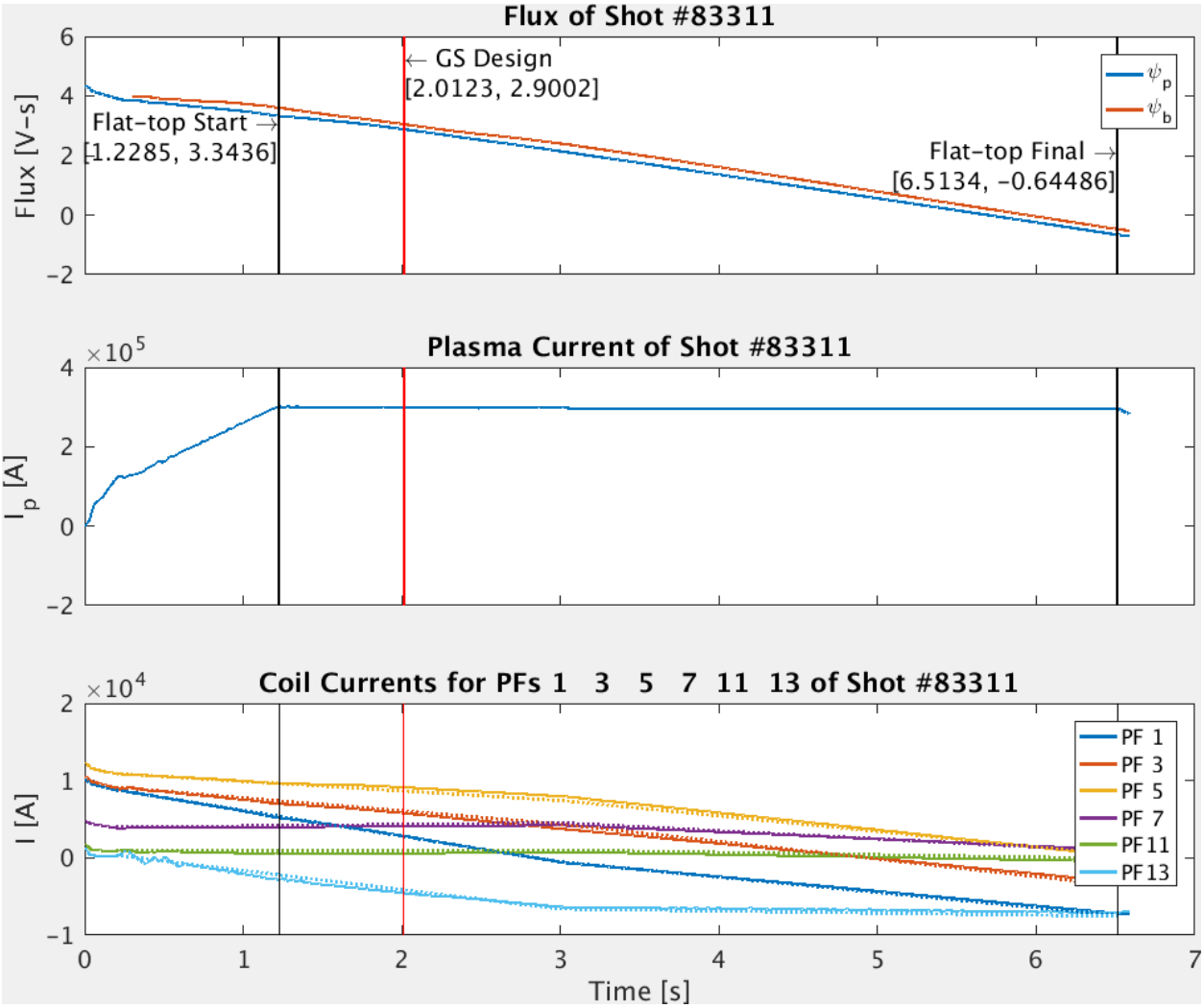


Figure 2.11: Flux progression during discharge and relationship with selected poloidal magnetic field coil currents. The bottom subplot shows solid lines for the measured coil currents and dotted lines for the designed coil currents. This figure is used both for explaining the design strategy concerning Equation (2.4) and is the result of shot 083311, this successful NTC discharge.

feedback control algorithms to smoothly transition the plasma from a circular shape to a fully diverted and elongated shape. The specific shape of the plasma is achieved in various ways, but all rely on the feed-forward current in the poloidal field coils. This work primarily focused on the use of the RZIP and ISOFLUX control schemes. The RZIP control simplifies the control system by assuming only two degrees of freedom i.e., movement on the $r - z$ plane. The main inputs needed from the equilibrium design are the initial feed-forward coil currents and the location of the magnetic axis. The targets used to create the designed equilibrium are also input into the PCS, but these are already known. The model assumes a current distribution for a specific plasma equilibrium and then only allows the plasma to move as a single rigid body in the vertical or radial directions [28]. The position is then regulated by adjusting the current in the PF coils to try and keep the magnetic axis at the desired location. The ISOFLUX control, taken from DIII-D, adds to the RZIP scheme by using specific control points on the boundary (points at which a diagnostic probe can directly measure the flux) and tries to keep the flux, which is calculated at each point using real-time EFIT, at all of these control points the same; greatly improving shape control [28]. Thus for both control schemes, the feedback control current for the shape position is superimposed on the feed-forward current for the desired shape. The sum of the two is the total current in the coil and is the reason particular feed-forward design constraints must be applied such that each coil has ample amperes of capacity remaining for the feedback control. Referring back to Figure 2.1a, a general design strategy of only allowing any given coil to use 89% of its capacity for feed-forward current is employed, the remainder is saved for feedback control. This design strategy can be seen in the top left subplot of Figure 3.1a. The upper and lower bounds are the "soft" limits of each coil and are only 89% of the "hard" or physical limits, see Table 2.4.

Chapter 3

Experimental Results

3.1 Design agreement

After completing all the steps in Chapter 2 and especially going through 17 versions of designs, each time fine-tuning one of the parameters, targets, or weights, the final equilibrium can be seen in Figure 3.1a and the key parameters are summarized in Table 3.1. The experimental results were gained after a total of five discharges were used to calibrate the PCS and produce the discharge (shot 083311) displayed in Figure 2.11 and Figure 3.1b. The agreement between the designed plasma parameters and the experimentally measured parameters are also shown in Table 3.1, along with the errors of each.

The large errors are not a concern as the designed equilibrium was for a specific flux and consequently falls into a different time-slice of the discharge than the selected time-slice of experimental results. The reason this time-slice of experimental results was chosen is that it shows the best results for the lower triangularity which is the main focus of this research. In the table, it can be seen that δ_L is more negative than the design, which is quite unexpected but very welcome. The parameters from this discharge will be used to create a new design that should be even more accurate with even lower δ_L. Furthermore, the κ and R_{ma} have an excellent agreement between the designed and experimental values. These are very promising results because κ has a strong relationship with

vertical drift. The greater the κ the more difficult it is to control the vertical position of the plasma. There was a concern that after augmenting the shape, the κ was near the limits of controllability for this novel NTC plasma. The agreement between the R_{ma} values is also encouraging as the observer used to estimate the horizontal position of the plasma needed to be fine-tuned during the preceding four discharges.

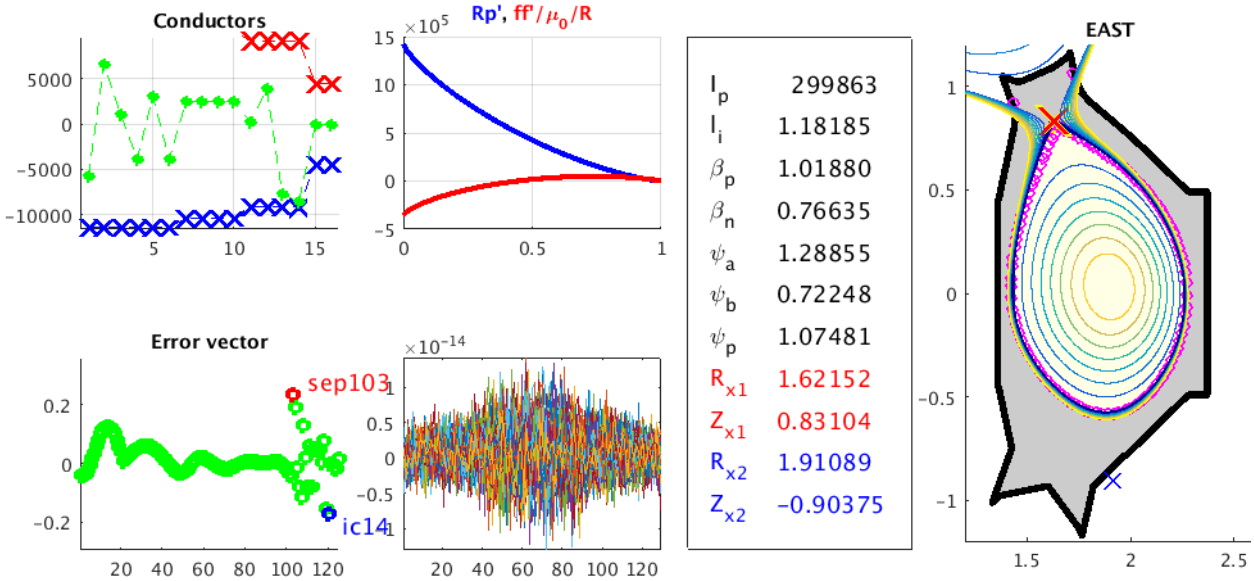
However, the experimental I_p is not as high as the designed I_p and this is a concern. Further qualitative investigation shows that because so much of the poloidal magnetic field coil flux was used in keeping a very negative δ_L the plasma current suffered. The vertical position of the magnetic axis, Z_{ma} is also of concern as this shows an error in the PCS. Normally, the vertical position is easier to control than the horizontal position, but the NTC has vertical stability issues in other experiments [16], [21], [24] and so it is to be expected that the vertical position would have a larger error. On the other hand, the discharge did not drift vertically enough to cause a disruption, and so further discharges are expected to be equally controllable.

Table 3.1: Summary of key parameters for final NTC design based on g170921

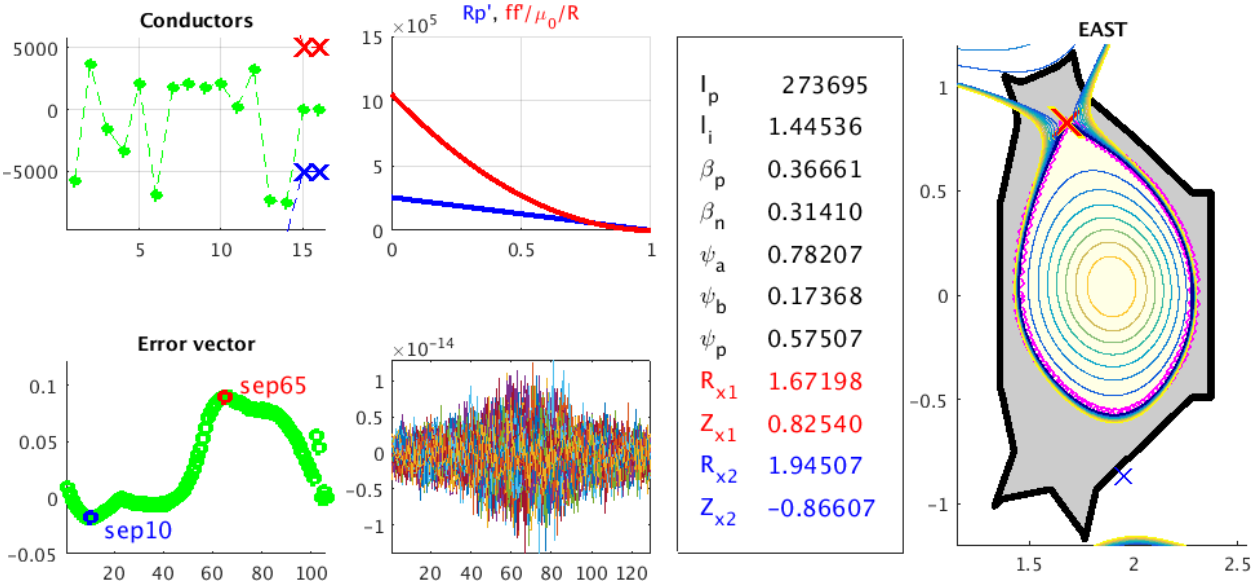
Parameter	Symbol	Units	GSDesign	Experiment	Error %
Plasma Current	I_p	[MA]	0.2999	0.2734	8.84
Internal Inductance	l_i	[H]	1.1818	1.4454	22.30
Poloidal beta	β_p		1.0189	0.3666	64.02
Elongation	κ		1.6670	1.6535	0.81
Lower Triangularity	δ_L		-0.0832	-0.0938	12.74
Magnetic Axis R	R_{ma}	[m]	1.8680	1.8945	1.42
Magnetic Axis Z	Z_{ma}	[m]	-0.0438	0.0400	191.32
Boundary Flux	ψ_b	[V-s]	0.7225	0.17368	75.96

In addition to the tabular results, it is helpful to plot the boundary points of the design and experiment for comparison. Qualitatively, the agreement is quite good and significantly better than expected.

The final check of this discharge is to examine where exactly the open field lines were incident upon the divertor plates. Figure 3.3 clearly shows that while the majority of the open field lines missed the critical areas, the strike points are very close. For future designs, the strike points will



(a) The final equilibrium design given by `gsdesign.m` based off of file g170921



(b) Reconstruction of the NTC shot 083311 on EAST

Figure 3.1: A comparison between the designed discharge and the reconstruction of the discharge 083311

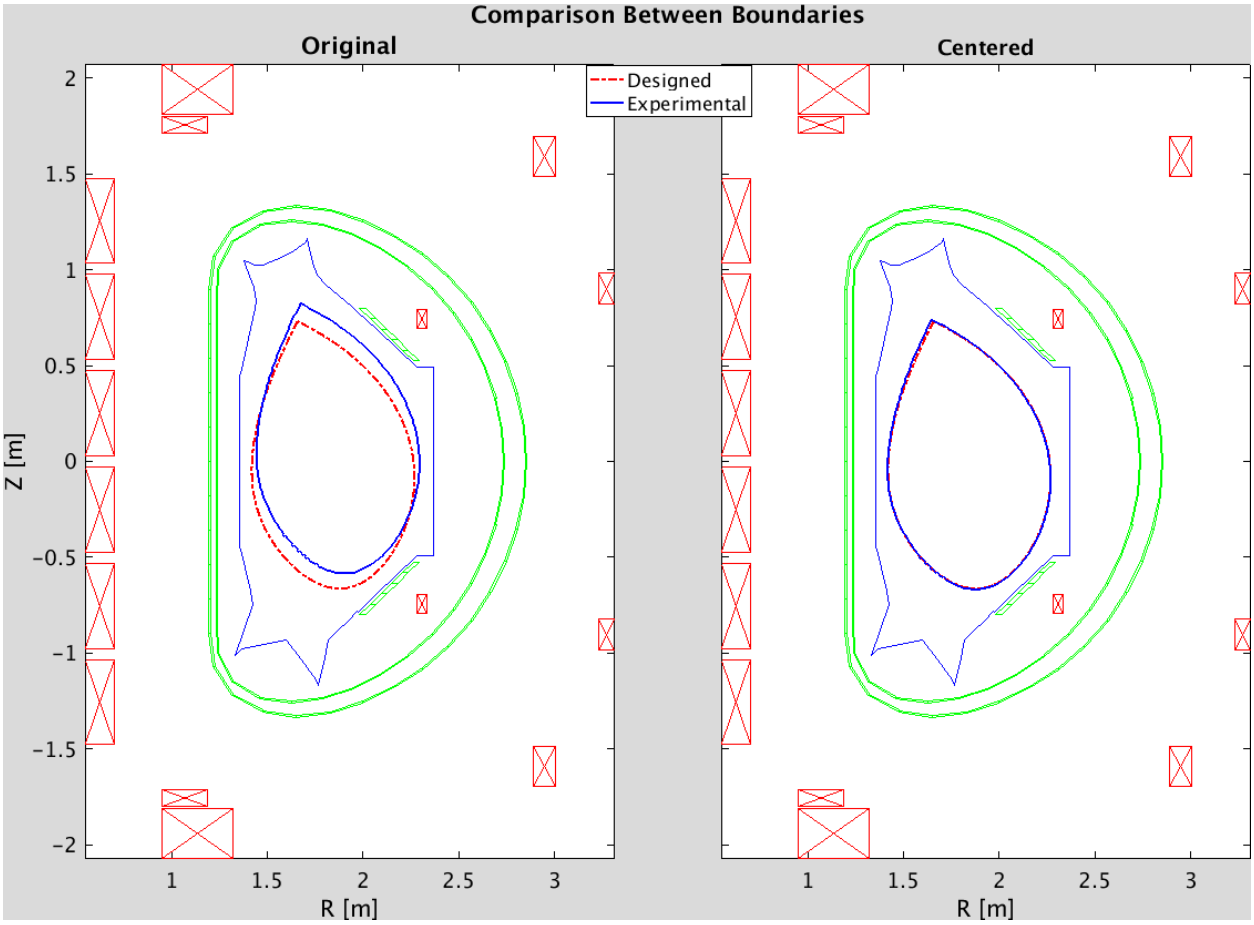


Figure 3.2: Plot of the boundary points of the design and experiment for discharge 083311

need to be moved farther up into the divertor. Moving the strike points deeper into the divertor is not just a goal of the NTC but is desirable for any fusion reactor design. This keeps the heat and particle flux that comes from the open field lines incident at an oblique angle effectively increasing the incident area and thus reducing the power density on the target plates.

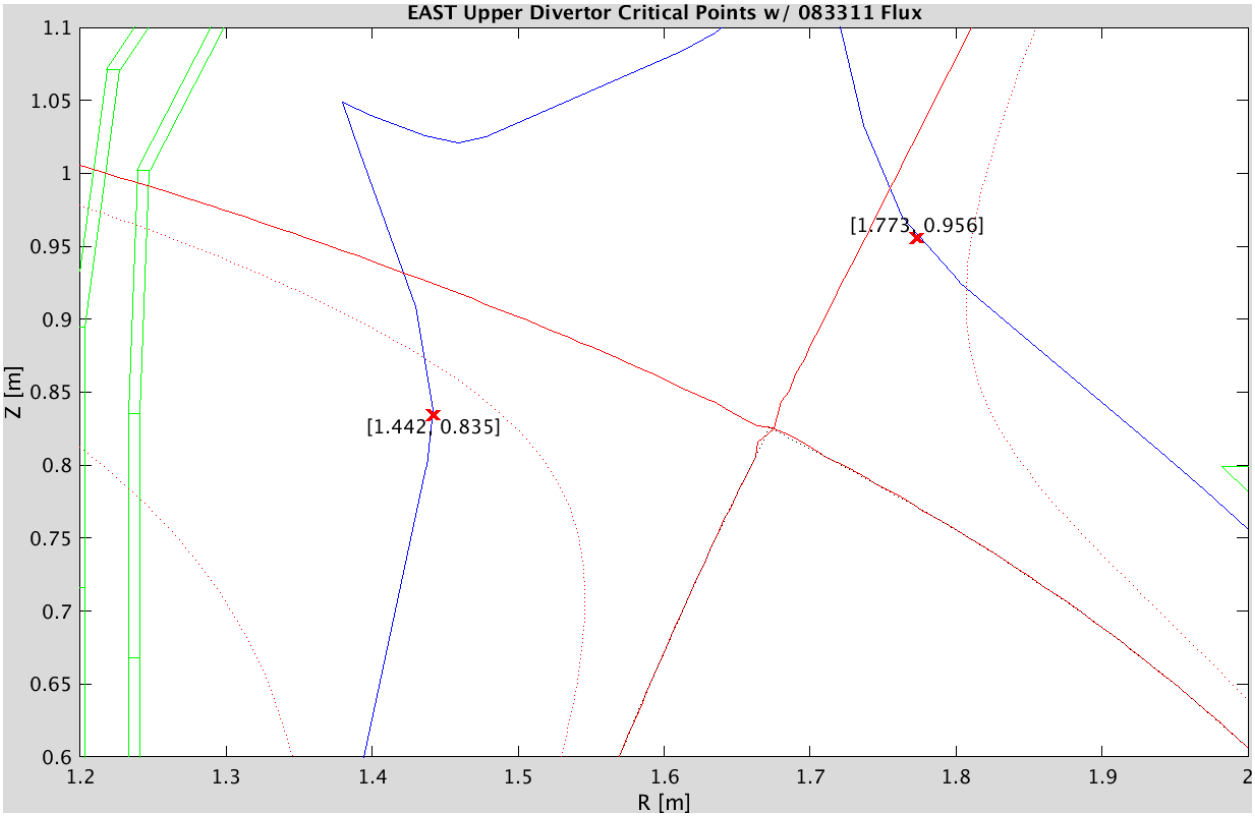


Figure 3.3: Closeup of the critical points in the upper divertor with the open field lines of discharge 083311

3.2 Limitations

The major limitation seen in these experimental results is largely from competing design requirements. The items listed below are the parameters in order of preference from a result standpoint, assuming the safe operation of the experiment.

- Strong negative-triangularity
- Coil limits - for safe operation and to maximize discharge duration

- Strike point locations
- Plasma current
- Plasma volume

Strong negative-triangularity, as stated above, requires the poloidal field coils to operate quite close to their limits, especially coils 13 and 14, as seen in Figure 2.11 (bottom subplot) and Figure 3.1a (top right subplot). This is understandable as coils 13 and 14 (see Figure 2.1a are closest to the right side of the plasma where coils 12 and 14 must work to keep the lower triangularity as negative as possible while coils 11 and 13 struggle to keep the upper right strike point away from the critical area of the divertor. In the future, to alleviate this problem, the target plates can be adjusted so that the seam between them is in a safer place. In fact, in the autumn of 2019, a full metal upgrade to EAST began. This will not only upgrade the divertors but the walls of the tokamak as well. This upgrade is expected to increase the heat-load the PFCs can handle as well as reduce the impurities that the PFCs release into the plasma.

The coil currents are certainly the crux of all the issues listed above. If the coils were perfectly superconducting, with infinite flux, then no problems would be had. However, the limitations exist exactly because this is not the case. The coils can only provide so many V-s and so this flux must be shared among the competing design requirements. Furthermore, the increased plating on the walls from the upgrade mentioned above will have increased eddy currents that will resist the changes in the magnetic field produced by the coils, effectively reducing the flux available for feedback control. However, a potential method to mitigate this is to get more accurate measurements of the passive conductors present in the magnetic fields. This plan is also underway during the upgrade. The control of the plasma is based largely on diagnostics that measure the magnetic fields *and* a model of the mutual inductance between the coils and passive elements. As of 2020, this model is still simple with many of the values being rough estimates (e.g., resistance between the power supply and the poloidal field coil) or some are even guesses (the resistivity between passive elements). Proposals to update this model have all been waiting for a time when the tokamak is disassembled

and so should be underway in late 2020. As a way to calibrate this model even better (many of the values change every time a passive element is added, subtracted, or moved) a proposal to add hardware to more accurately measure the induced voltage at each coil such that the coils themselves can be used to measure the mutual inductance. Imagine sending a waveform, a simple sawtooth, for example, through one coil and using all the other coils as well as the magnetic diagnostics to measure the field produced. Then repeated, not only for each coil but for every combination of the coils. Using this method, an overdetermined system of the coils and passive elements could then be solved using a least-squares method. The proposal for this upgrade is still under consideration.

The limitations due to the strike point locations have already been discussed, leaving only the final two items. The plasma current is the more important of the two. A higher plasma current results in a higher B_ϑ which increases the confinement of the plasma. As discussed above, the poloidal field coils struggle to stay within their limits to achieve the NTC and so the plasma current suffered. However, if the proposal mentioned in the preceding paragraph were implemented then the magnetic diagnostic measurements would hypothetically be more accurate making the observer for the plasma center more accurate and thus the plasma would be more controllable. This would allow the coils to operate closer to their limits for longer without fear of disruption or not having enough flux in reserve for feedback control and thus able to increase the plasma current. This would also work well for the plasma volume issue as the boundary points would then be more accurately measured in real-time and more easily controlled. This means the safety margin between the LCFS and the limiter can be safely decreased, allowing for a larger overall plasma.

We wait in hopeful anticipation of the passive voltage measurement upgrade for the poloidal (and possibly the toroidal) field coils to be approved so that a team of clever and feisty graduate students can cut their proverbial teeth on it.

Chapter 4

Plasma Response Simulation

4.1 Introduction

This section gives a detailed demonstration of `east.slx`, a `Simulink` model, to produce a simulated shot using the EAST PCS. One of the aims of this simulation, and by extension this section, is to allow students and scientists alike to test out their proposed shots in a simulated environment and thereby work out any bugs/mistakes with their design/configuration/diagnostics/etc. To achieve this, the majority of the heavy lifting is done by `gspert.m` which can be used separately, though it will not be covered in this thesis. Designing an equilibrium of a shot can be done with `gsdesign.m` and has already been covered in Chapter 2. The general procedure of this simulation is to use the existing PCS as is and use `Matlab` to provide all the same feedback that the EAST machine would provide. To do this, the `east.slx` needs to match the plasma response with reasonable accuracy, which it does as of the spring of 2020 and is continuously improving. At the moment, `simpcs` in conjunction with `east.slx` is less forgiving than the actual EAST machine in most cases. That is to say, if a discharge will not run on the `simpcs`, then it still might run on the EAST machine. However, if the discharge does run on the `simpcs` then it will almost certainly run on the EAST machine.

Note that `simpcs` will refer to all the software used to simulate a shot at EAST while `east.slx`

refers only to the Simulink model in Matlab. The background necessary for this section is the same as that of Section 2.3.

4.2 Starting File(s) and information

4.2.1 File(s)

This section will discuss the inner workings of three files. Two of which can be copied from the TOKSYS library and modified while the last one will need to be written by the user just as those in Section 4.2.1 were written. It is recommended that the user create a directory for the EAST simulations. For this section, create the following EAST-sim directory located at the path /home/ASIPP/<user>/EAST-sim/ where <user> is the user's home directory. Start by opening a Linux [Terminal], log into data-server3, and use the following commands to copy two files from TOKSYS to the EAST-sim directory. If the user is unsure how to open a Linux [Terminal], refer to Appendix B.

Listing 4.1: Copying files from TOKSYS to the user's EAST-sim directory

```
1 cp /project/builds/TOKSYS/2019-09-05_00-24-57_build366/tokamaks/
    east/sim/gsevolve/setup_east.m /home/ASIPP/<user>/EAST-sim/
2 cp /project/builds/TOKSYS/2019-09-05_00-24-57_build366/tokamaks/
    east/sim/gsevolve/simsettings.m /home/ASIPP/<user>/EAST-sim/
```

At the time of writing, the source file path above is the current path. In the future, this path will change and so the source file path could also be written as /project/builds/TOKSYS/current/tokamaks/east/sim/gsevolve/setup_east.m. However, the latest current directory may have updated simsettings.m and setup_east.m which might not match those in this section and may not work smoothly with the steps in this section.

4.2.2 TOKSYS

Now that the necessary `Matlab` files have been copied, one should move on to creating a shell script called `simpcs.bashrc`. This script is run from the Linux `[Terminal]` window before `Matlab` is run and sets the global variable `GATOOLS_ROOT` to a specific path as well as a few other aliases. For `east.slx` to work properly for what follows, this variable needs to be set to the build from 2019-09-05. As updates are made to the TOKSYS there is no guarantee that this tutorial will work for later builds of TOKSYS. It is recommended that this section is completed before using a later build of TOKSYS. For this, look at the following bash script named.

Listing 4.2: bash script used after logging into a server

```
1  # simpcs.bashrc
2
3  cd /home/ASIPP/daw/EAST-sim
4
5  # Set paths
6  export MATLABPATH=/project/builds/TOKSYS/current/startups:
       $MATLABPATH
7  export GATOOLS_ROOT=/project/builds/TOKSYS/current
8  export PATH=/project/builds/anaconda2/bin:$PATH
9
10 # User specific aliases and functions
11 alias ls="ls -alh --color"
12 alias matlab="matlab2017a -softwareopengl"
13 alias simpcs="/pcshome/pcs/builds/pypcs_shared/pcshost/current/
       tokrun nosim --pcs-root=/pcshome/pcs/builds/PCS/east/pcshost/
       master/current --gui"
```

Line 3 changes the working directory, lines 4, 5, and 8 set aliases that make executing certain commands easier, while lines 6, 7, and 9 set the necessary paths for `simpcs` and the `Matlab` scripts to execute properly.

4.3 Running the Matlab Scripts

Begin with opening a Linux [Terminal] in the location of the EAST-sim sub-directory. Execute the commands after each "$" without a preceding space and the output should match the output below.

Listing 4.3: What the [Terminal] should look like after starting Matlab

```
bash-4.1$ ssh -X data-server3
<user>@data-server3's password:
Last login: Wed Apr  8 06:40:02 2020 from 202.127.205.60


************************************************************

 Welcome to EAST PCS Data Server

************************************************************

-bash-4.1$ source simpcs.bashrc
-bash-4.1$ matlab
MATLAB is selecting SOFTWARE OPENGL rendering.
```

At this point, the Matlab GUI should open. Again, it should be noted that some if the quick key commands such as ctr+c and ctr+p do not behave the same way when using Matlab on the server as they do on a PC. For a remedy, please see Appendix E.

4.3.1 sim_start.m

For convenience, create a start-up script that will run several other scripts to load everything east.slx needs to work properly.

Listing 4.4: `sim_start.m` start-up script

```
%  Used to set file paths, run toksys_startup, east_startup, simsettings,
%  and setup_east. All unnecessary variables are cleared at the end.
%
%  WRITTEN BY:  David Weldon  ON        2020/03/30
%
%  MODIFICATION HISTORY:
%       2020/04/04  Cleaned up the code to make it more readable for a tutorial
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

close all
clear all
clc

disp('**********sim_start**********')

% Get the basics started up
disp('**********toksys_startup**********')
toksys_startup
disp('**********east_startup**********')
east_startup
% settings for simserver
disp('**********simsettings**********')
simsettings
% run the setup for the east sim
```

```
26 disp('**********setup_east**********')
27 setup_east
28
29 % set some defaults for figures
30 set(0,'DefaultAxesFontSize',15)
31 set(0,'DefaultFigureColor',[255 253 224]/255)
32 set(0,'DefaultFigureColormap',jet);
33 set(0,'DefaultAxesCreateFcn','zoom on')
```

Lines 11-13 are optional, in order they: close all figure windows, clear the *Workspace*, and clear the *Command Window*. Lines 15, 18, 20, 23, and 26 display in the *Command Window* the script that is running to make it easier to see the progression of the startup. Lines 19, 21, 24, and 27 execute the corresponding scripts. Lines 30-33 set some default values for plotting figures.

To create and run this script, go to the `Matlab` GUI and in the top right corner select "New Script" as seen below in Figure 2.3. Copy and paste the code from Listing 4.4. Be sure to check for proper formatting as copy and pasting from this PDF file might not work as expected. Once it is copied, then save it as `sim_start`.

4.3.2 `simsettings.m`

Before running this script, however, set the shot number in the `simsettings.m` script to 77218 (the discharge that will be used throughout this section) and while doing so, add a few other details as well. The `simsettings.m` script that was just copied from the TOKSYS directory in Section 4.2.1 should be changed slightly to match the code in Listing 4.5.

Line 5 is the shot number that will be loaded into the PCS and simulated. Line 13 is optional and can be set to any one of the directories shown in line 11. This tells `Matlab` where in the MDSplus tree to look for the data of the shot number in line 5. Line 18 needs to match the stop-time that is used in the PCS NEXT SHOT and will be discussed later. Lines 23-25 are needed for `setup_east.m` which will be addressed next.

Listing 4.5: `simsettings.m` simulation settings i.e. shot number and MDSplus directory to search

```
% Create a copy of this file in the directory from which matlab is run
% Use matlab version 2017a:

% oldshot is default source for data that are not otherwise specified
oldshot = 77218;

% set efit source to one of the following. if the source you chose is not
% available then setup_east.m will attempt to retrieve data from the
% followin in the order presented until data is found. If no data is found
% then setup_east will exit with an error.
% 'EFITRT_EAST', 'EFIT_EAST', 'PEFIT_EAST', 'PEFITRT_EAST'

% efit_source = 'EFIT_EAST';



% stop_time should match time-to-stop in the PCS under Bookmarks-> operating setup data
stop_time = 2;


% ------------------------------------------------------------------
% Creating structure simset that is read by setup_east (do not edit)
```

```
23 simset = struct(...
24   'oldshot',      oldshot,...
25   'stop_time',    stop_time);
```

4.3.3 setup_east

Only lines 55-105 are displayed as the other lines will not be modified. Starting with lines 55-58 delete the line setting `efit_source` to match that of Listing 4.6. Then move down to lines 77-101. It is easiest to copy and past from Listing 4.6 and then reformat to match it exactly. In short, the line that defines `efit_source` has been taken from lines 55-58 and moved it to an `if` statement in lines 77-101 that checks for the existence of the `efit_source`. This variable is optionally defined in `simsettings.m` in line 13 as shown in Listing 4.5. However, that line is commented out which means that `setup_east` will iterate through the list of each possible `efit_source` until the data is found or exit with an error if it is not found. For this example, the source will be found and it will be from `EFIT_EAST`, however for other shots the source may be different and will need to match the source chosen in Section 4.4.2. This is done with the `while` loop in lines 83-97 of Listing 4.6. Information is displayed to the *Command Window* in `Matlab` to indicate which sources are tried and which one is chosen.

Listing 4.6: `setup_east.m` simulation settings i.e. shot number and MDSplus directory to search

```
55 tok.tokamak = upper(tok.tokamak);
56
57 rzero = 1.6955;
58 shot = simset.oldshot;
59
60 % Data and commands from the old shot
61 getallolddata = 1;
62 if getallolddata
63   disp(['Reading pcsdata from shot ' num2str(shot)])
```

```
  pcsdata = east_pcsdata(shot);

  disp(['Reading pcs commands from shot ' num2str(shot)])
  pcscom = east_pcscom(shot);
else
  pcsdata = [];
  pcscom = [];
end

if verbose
  disp(['Reading efits from shot ' num2str(shot)])
end

if ~exist('efit_source', 'var')
    efit_source = {'EFITRT_EAST', 'EFIT_EAST', 'PEFIT_EAST', '
       PEFITRT_EAST'};
    disp('**No efit source specified in the simsettings.m file.**')
    disp('**These sources will be attempted in the order specified:
       ');
    disp(efit_source)
    c = 1; exit = 0;
    while exit == 0
        try
            disp(['**Trying to find data in source: ' efit_source{c
               } '**'])
            [eqs, eqtimes] = east_efits(shot,efit_source{c},tok);
            disp(['**Using data found in source: ' efit_source{c} '
               **'])
            exit = 1;
        catch ME
            disp(ME)
            c = c + 1;
```

```
92          end
93          if c > length(efit_source)
94              exit = 1;
95              disp(['**No data found for shot ' num2str(shot)])
96          end
97      end
98  else
99      [eqs, eqtimes] = east_efits(shot,efit_source,tok);
100     disp(['**Using data found in source: ' efit_source '**'])
101 end
102
103 k = true;
104 for i = 2:numel(eqs)
105   k(i) = eqs(i).betap > 0;
```

4.3.4 sim east

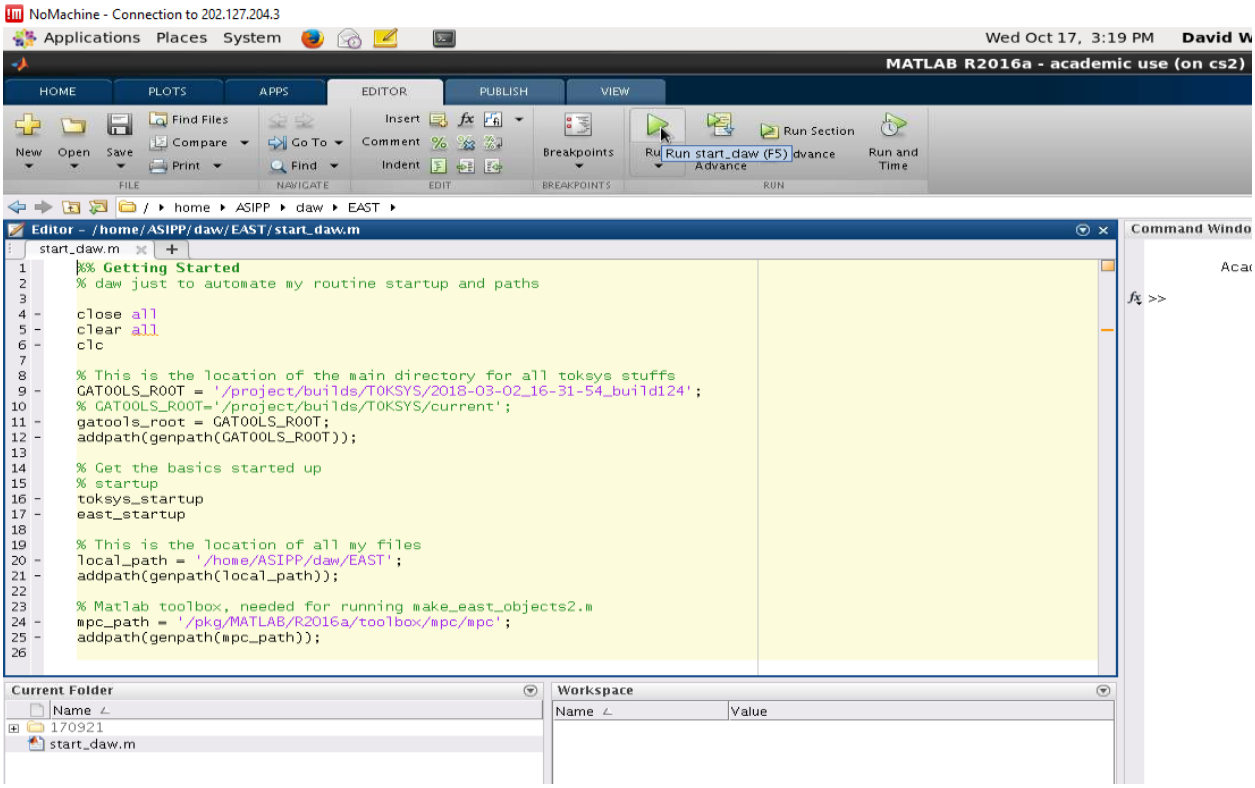


Figure 4.1: Running a `Matlab` script

With all three `Matlab` scripts ready, the `simsettings.m` and `setup_east.m` scripts can be closed and the `sim_start.m` script can be executed by clicking on the large green arrow near the top center of the `Matlab` GUI. Alternatively, the `F5` key on the keyboard will work also. Once executed the GUI should be similar to Figure 4.1. The first time this script is run, three files will be created and added to the directory and the *Command Window* may show some warnings and errors when looking for these files. This can take several minutes.

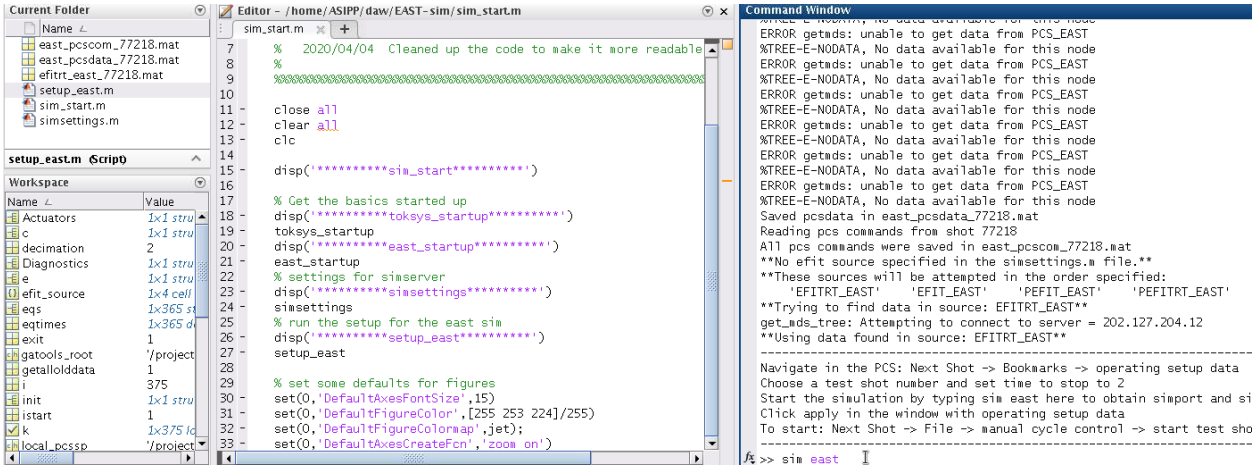


Figure 4.2: The results of running the `sim_start.m` script for the first time

If the script executes without serious error, then the final output to the *Command Window* should be exactly like that of Figure 4.2. Also shown in the figure is the last `Matlab` command to be run: `sim east`. Type this in as shown and hit *enter*. This final command will start up the simulation and wait for the PCS to begin interacting with it.

To see the `Simulink` file `east.slx` instead of just running it, then replace the `sim east` command with simply `east`. This will open the `Simulink` model, as shown in Figure 4.3. This model can be run by clicking the green arrow button near the top center of the `Simulink` window. This will start the simulation and should yield the same results as running the `sim east` command in the *Command Window*. However, if the simulation seems to end abruptly or unexpected behavior occurs, then revert to using the `sim east` command in the *Command Window*.

Now that the simulation is running, look at the Linux `[Terminal]` window shown in Listing 4.7. Take note of Line 19 as it contains the most pertinent information for the next section which will

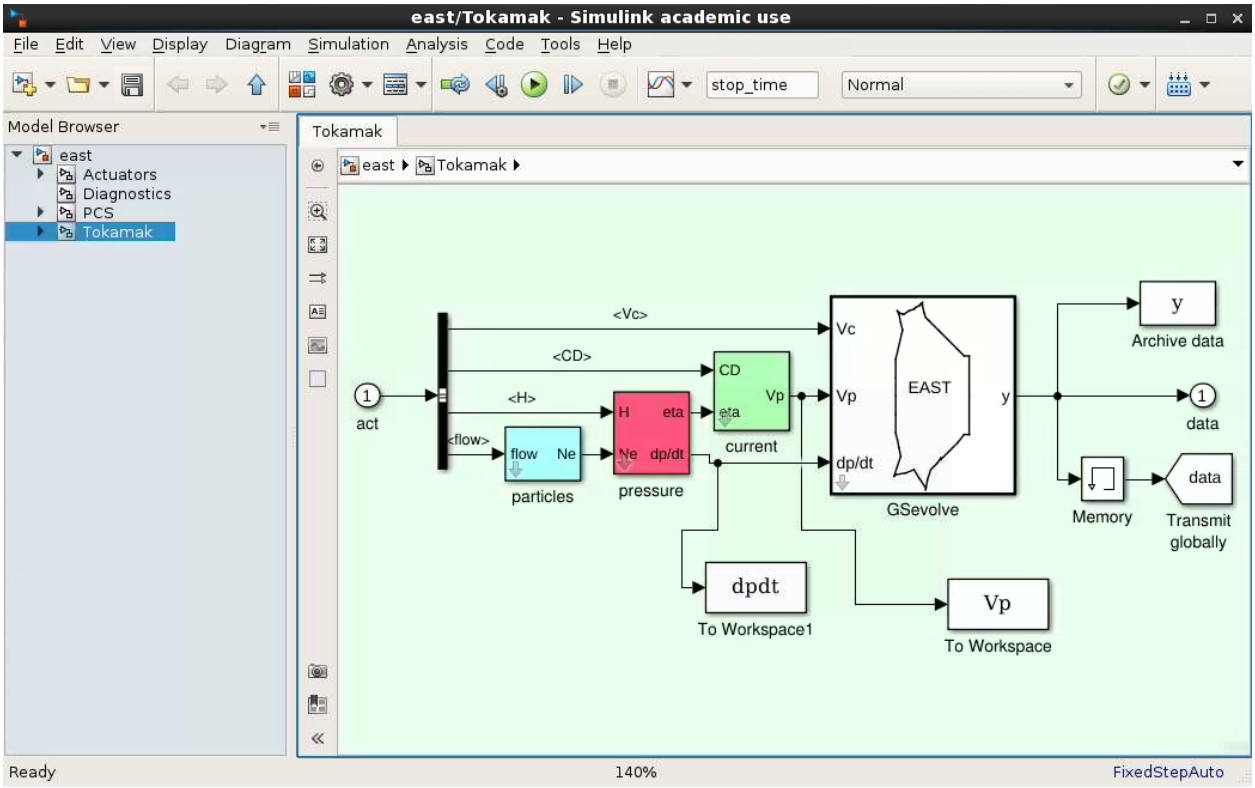


Figure 4.3: The `Simulink` model simulates the behavior of the EAST machine, opened by executing `east` in the *Command Window*

give instructions on how to open and configure the simulated PCS.

Listing 4.7: Information to be entered into the PCS is shown in the same Linux `[Terminal]` used to start `Matlab`

```
1  bash-4.1$ ssh -X data-server3
2  daw@data-server3's password:
3  Last login: Wed Apr  8 08:13:41 2020 from 202.127.205.60
4
5
6  ************************************************************
7
8   Welcome to EAST PCS Data Server
9
10 ************************************************************
11
```

```
-bash-4.1$ source simpcs.bashrc
-bash-4.1$ matlab
MATLAB is selecting SOFTWARE OPENGL rendering.

*****************************************
SIMSERVER ready and waiting for start of shot

simport = 53514, simhost = data-server3
num rt cpus = 7
master cpu num = 1
data will be given to cpus in following order: 1, 2, 3, 4, 5, 6, 7
simulation will start at 0.000000
cpu 1 cycle time = 100 ** MASTER CPU
cpu 2 cycle time = 500
cpu 3 cycle time = 4000
cpu 4 cycle time = 100
cpu 5 cycle time = 1000
cpu 6 cycle time = 100
cpu 7 cycle time = 500
cpu 1 command delay = 100
cpu 2 command delay = 500
cpu 3 command delay = 4000
cpu 4 command delay = 100
cpu 5 command delay = 1000
cpu 6 command delay = 100
cpu 7 command delay = 500

starting execution
*****************************************
```

4.4 Running `simpcs`

Begin this section without closing any window or changing `Matlab` in any way. Instead, simply open another Linux `[Terminal]`, log into `data-server3`, and run the `simpcs.bashrc` script as before.

4.4.1 Starting `simpcs`

Now instead of starting `Matlab`, execute the command `simpcs` as shown in Listing 4.8.

Listing 4.8: Executing `simpcs` in a new Linux `[Terminal]`

```
1  bash-4.1$ ssh -X data-server3
2  daw@data-server3's password:
3  Last login: Thu Apr  9 03:22:20 2020 from 202.127.205.60
4
5
6  ************************************************************
7
8   Welcome to EAST PCS Data Server
9
10 ************************************************************
11
12 -bash-4.1$ source simpcs.bashrc
13 -bash-4.1$ simpcs
```

This will cause the *IDL* window to open. Simply click on the "Click to Continue" button as shown in Figure 4.5a which will cause another window, called the *Wave*, to open as is shown in Figure 4.5b. Once the *Wave* window is open go to *Control* > *next shot* to open the *NEXT SHOT* window, as shown in Figures Figure 4.5b and Figure 4.4.

This brings us to the *Load shot* window shown in Figure 4.6a. Enter in the shot number 77218 and then click "load all categories". This will open a *TAKE NOTICE!* which gives some warnings.

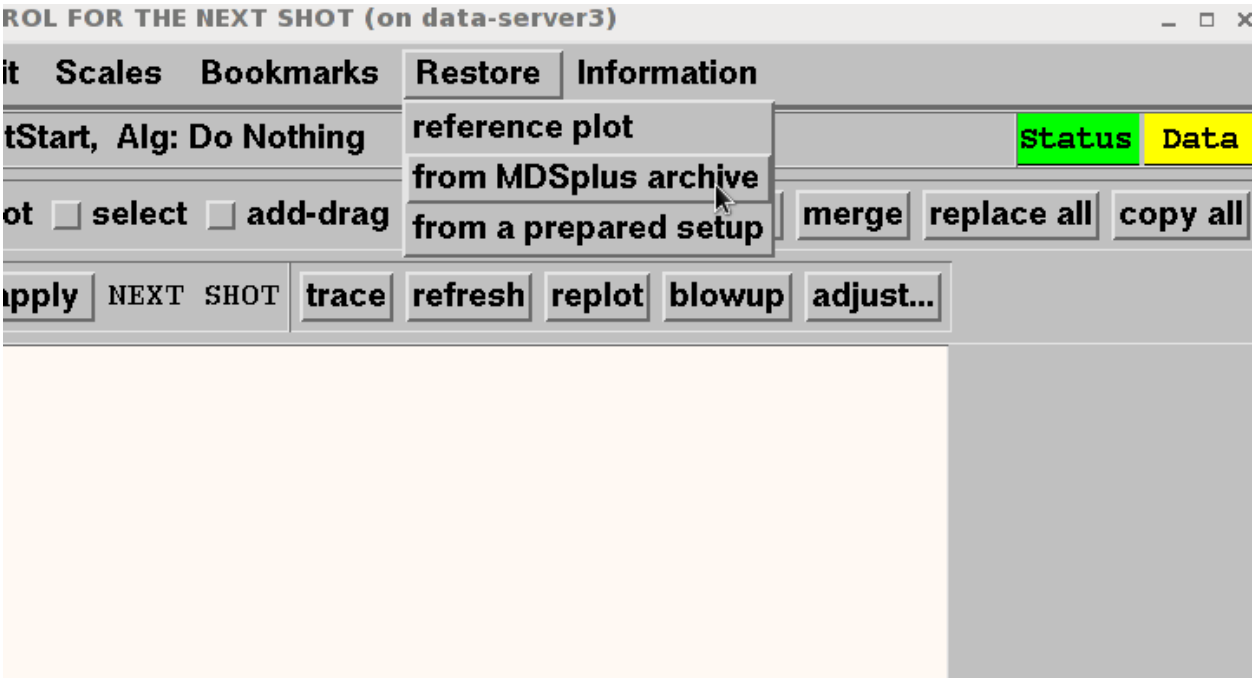
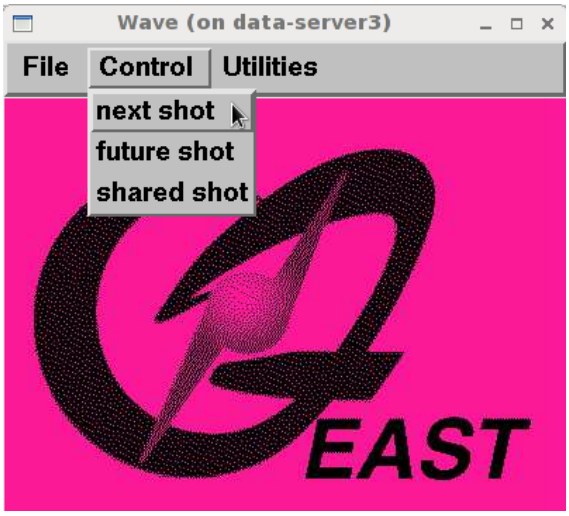


Figure 4.4: Restoring a previous shot in the *NEXT SHOT* window

(a) Starting the PCS requires use of IDL



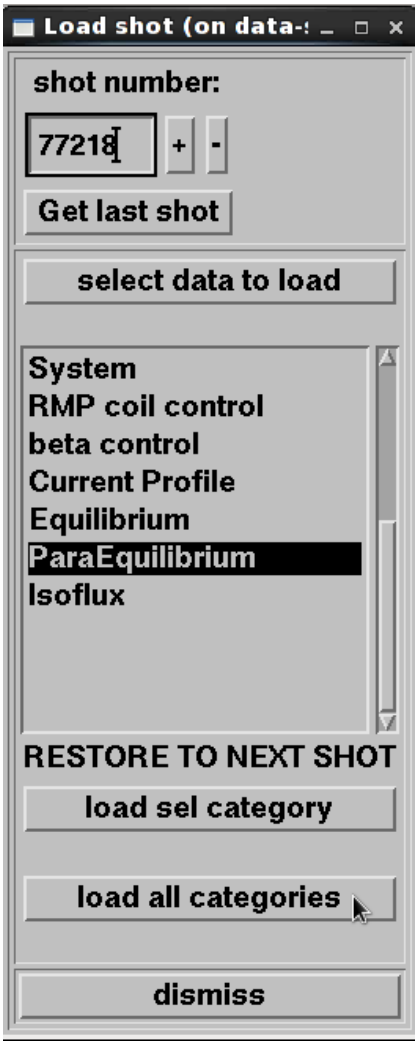
(b) Opening the *NEXT SHOT* window

Simply click the "dismiss" button at the bottom of the window which will then open another window titled *Skipped Items From Restores*. Click "load phase: ShotStart/Data Acquisition" which will lead to another *TAKE NOTICE!* window, which can also be dismissed. This will cause the *Skipped Items From Resotres* window to change slightly and add more options. Click on each option starting with "load data item: Snap setups/efit/Equilibrium" one by one until they have all been loaded, then click the "dismiss" button. After the first option is clicked, the space-key can also be used to click on each subsequent option. Finally, dismiss the *Skipped Items From Restores* and the *Load shot* windows.

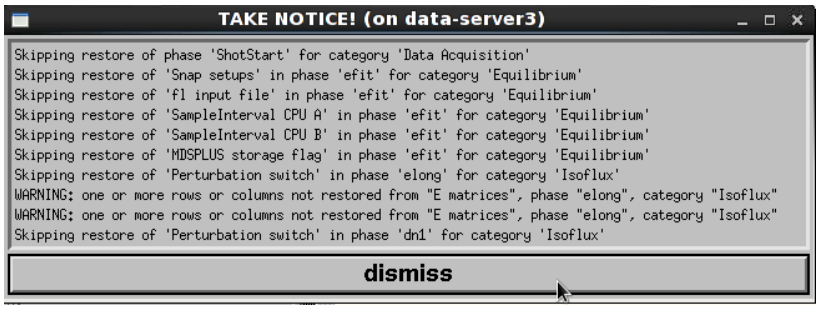
4.4.2 Configuring the Shot

Now that the shot is loaded, change the real-time reconstruction from PEFIT (Parallel EFIT) to rtEFIT (real-time EFIT). As mentioned in Section 4.3.3, this source will need to match the source that `Matlab` found. To do this, turn once again to the *NEXT SHOT* window in the left pane and follow the sequence *Ctgy* > *ParaEquilibrium* > *Sub* > *EQUIL options* > *Equilibrium Source*. This sequence is shown more explicitly in Figure 4.7. The next step is to return to the top of the *NEXT SHOT* window and from there select *Bookmarks* > *operating setup data* which will open a new window titled *Operating setup data*. Enter in the information as is displayed in Figure 4.8 except for the information noted earlier in Listing 4.7. Set *Operational mode* > *simulation test* which is the mode for working with the `east.slx`, the "time to stop" was set in the `simsettings.m`. The "simserver host:" and "simserver port:" come from the Linux `[Terminal]` from which `Matlab` is run, Listing 4.7 line 19. Finally, the "Number of samples:" should be set to 10,000. All other data does not need to be changed and can be left to whatever default the *Operating setup data* had originally. Click "apply" at the bottom of the window and then "close".

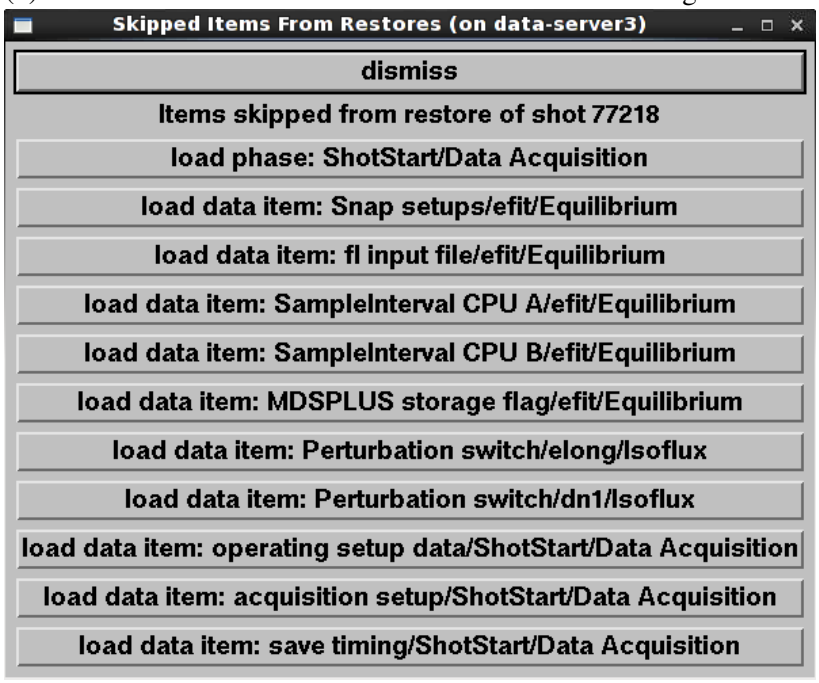
The last step to run a shot is to go to the *NEXT SHOT* window then select *File* > *manual cycle control* which will open a window by the same name. From here ensure that the message reads "waiting for a shot" then click "start test shot" which should change the message in the window to "locked out, unlock possible" as shown in Figure 4.9a. Shortly after that a new `Matlab` figure titled



(a) *Load shot* window, enter in shot number then select "load all categories"

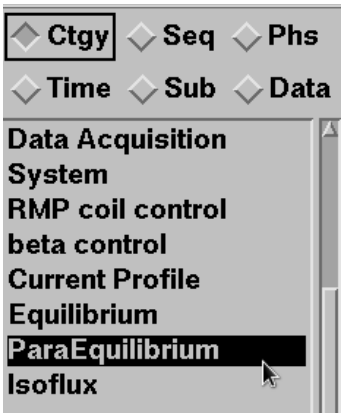


(b) *TAKE NOTICE!* windows can be dismissed after reading the notice

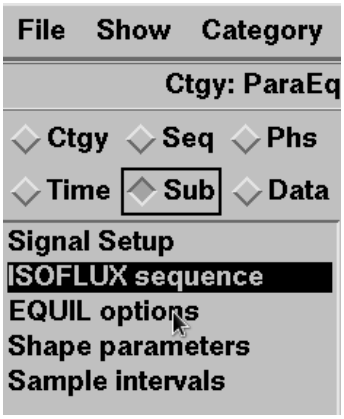


(c) *Skipped Items From Restores* window needs all "load..." buttons clicked

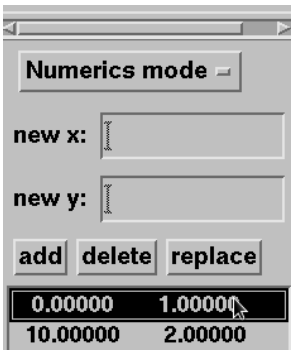
Figure 4.6: Steps and windows for restoring a shot



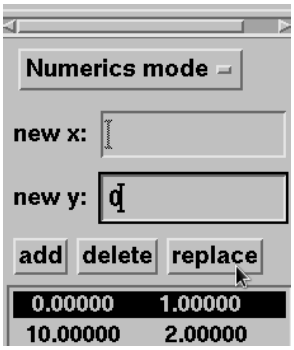
(a) Step 1. *Ctgy > ParaEquilibrium*



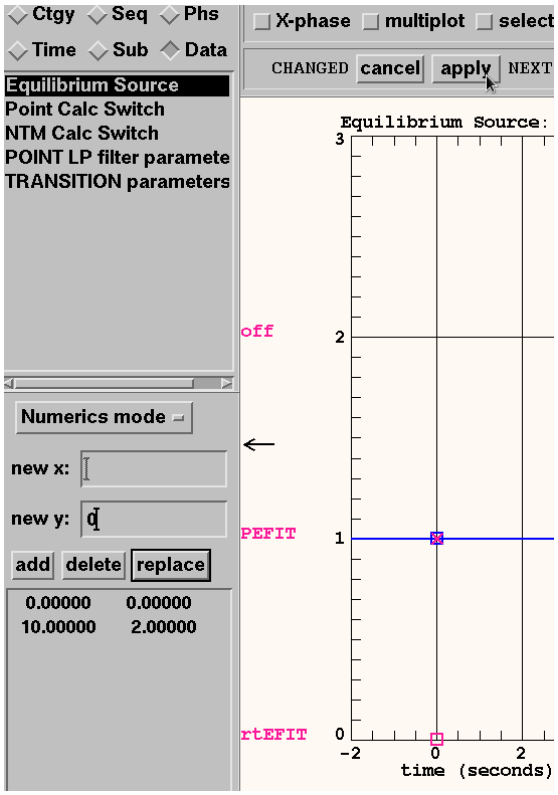
(b) Step 2. *Sub > EQUIL options*



(c) Step 3. *Equilibrium Source > 0.00000 1.00000*



(d) Step 4. In the *new y:* window enter *0* then click "replace"



(e) Step 5. On the graph, a square pink marker should indicate the new coordinate that was entered in the previous step. Just above the graph on the left side, click "apply"

Figure 4.7: Steps and windows for changing the EFIT source

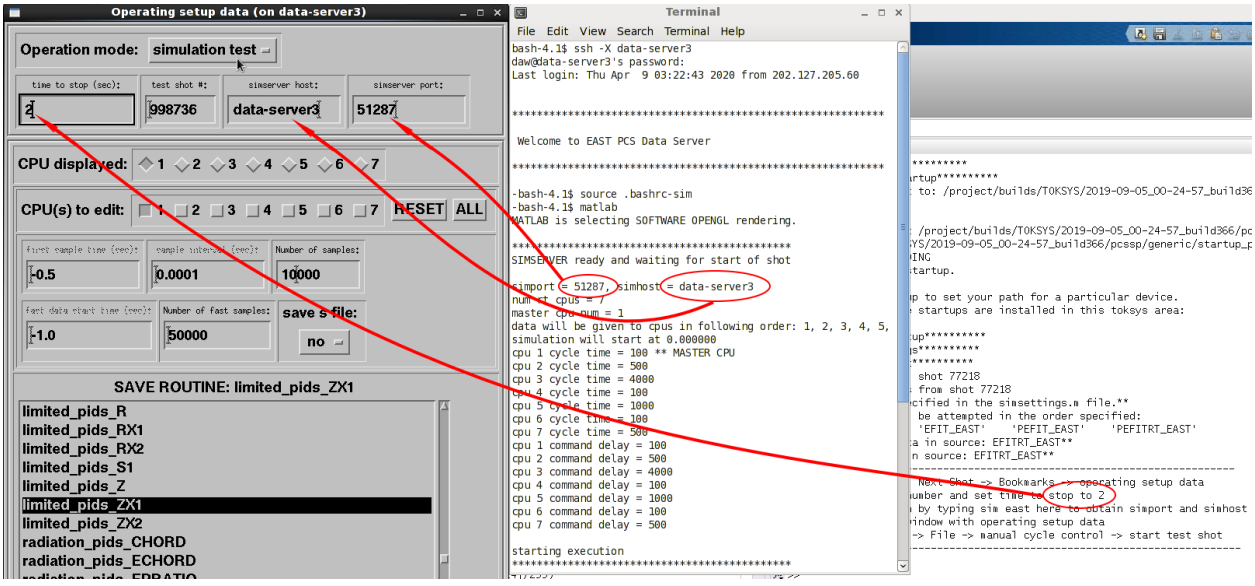


Figure 4.8: Entering the pertinent information from `Matlab` into the *Operating setup data* window

GSevolve simulation should open showing the progression of the plasma flux surfaces as it solves each equilibrium for each time step using `GSevolve`, shown in Figure 4.9b.

4.5 Post-simulation

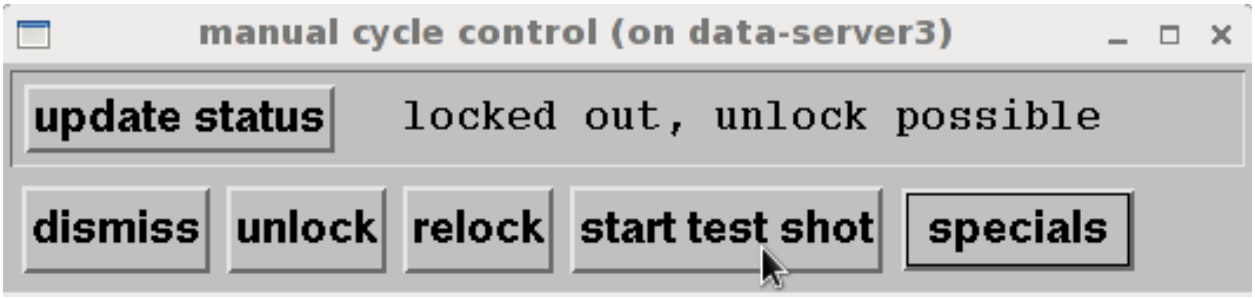
If any errors occur, then the log file which is located by looking at the *Wave* window then *Utilities > view pcs log* should hold greater details of the errors. This can give clues as to what is wrong with the PCS and simulation.

After or even during the simulation another Linux `[Terminal]` can be opened but instead of logging into the `data-server3`, log into the `csX` server where `X` can be 1, 2, 3, or 4. Then issue the command `eastviewer` as shown in Listing 4.9 which should bring up the window *EASTViewer on EAST*. Then select *EAST mds-server* and enter in the "shot:" then click "update" which will then populate the "Tree" list under the "tree and time" section. Since the simulation is based on rtEFIT, choose "efitrt_east" as shown in Figure 4.10. Now, select a time to view which will open another window: *Plasma Equilibrium*. This window, shown in Figure 4.11, gives the reconstructed shape and data for the shot for each time-slice and can be compared with the simulation as the simulation steps through time.

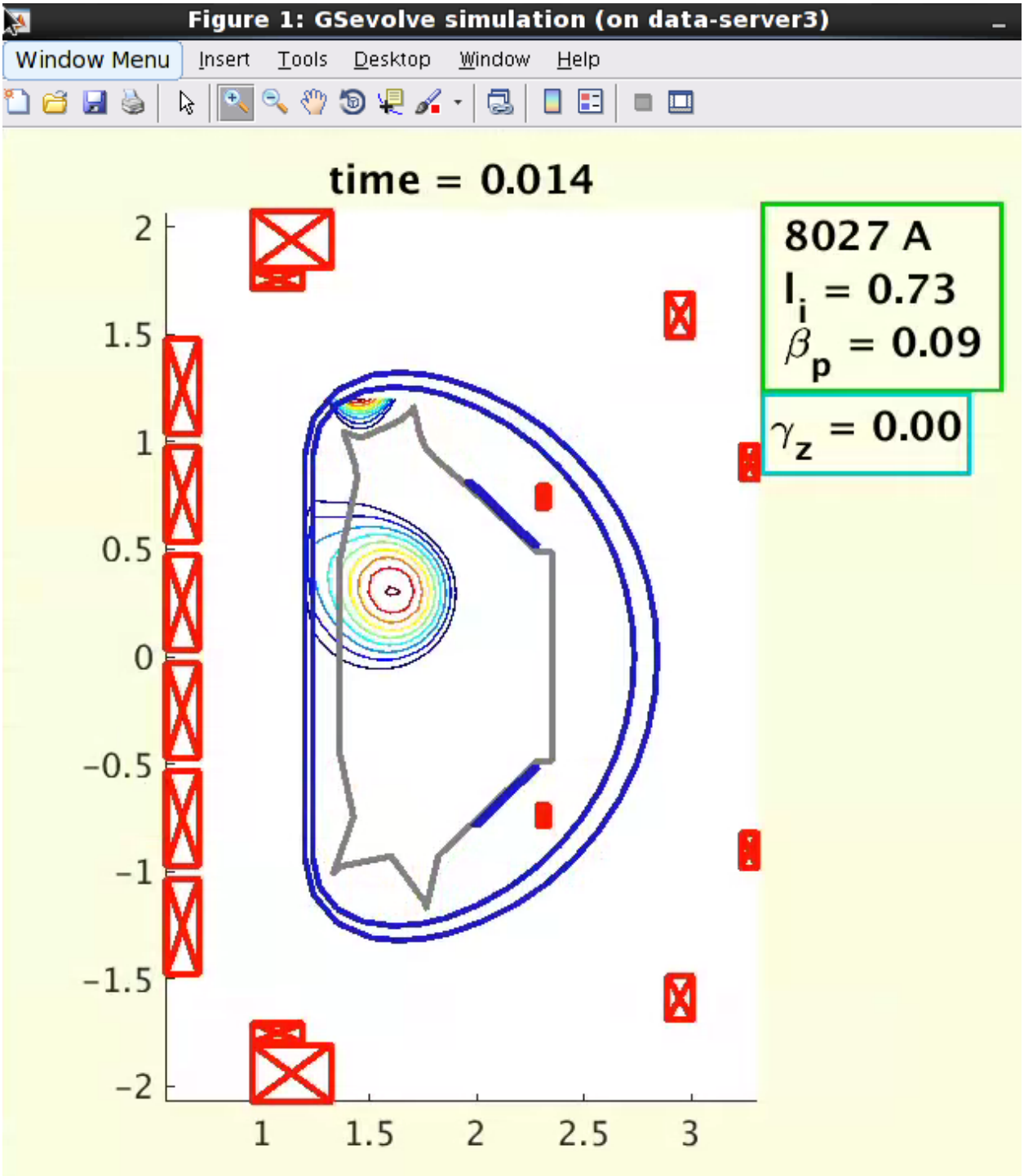
Finally, the simulation can be checked using the following command executed in the `Matlab` *Command Window*: `viewtime=1; check_east_sim`.

Listing 4.9: Viewing the simulation results in the `Matlab` *Command Window*

```
1 bash-4.1$ ssh -X cs2
2 daw@cs2's password:
3 Last login: Fri Apr 10 03:15:51 2020 from 202.127.205.60
4
5
6 ************************************************************
7
```



(a) Select *NEXT SHOT* > *File* > *manual cycle control* > *start test shot* to begin the test shot.



(b) The *GSevolve simulation* window shows the progression of the simulated plasma.

```
 Welcome to EAST Computing Server 2 (CentOS6.7-64bit)

 Parallel and high performance computing are prohibited!

 Please pay attention to resources usage or be banned!

************************************************************

-bash-4.1$ eastviewer
```

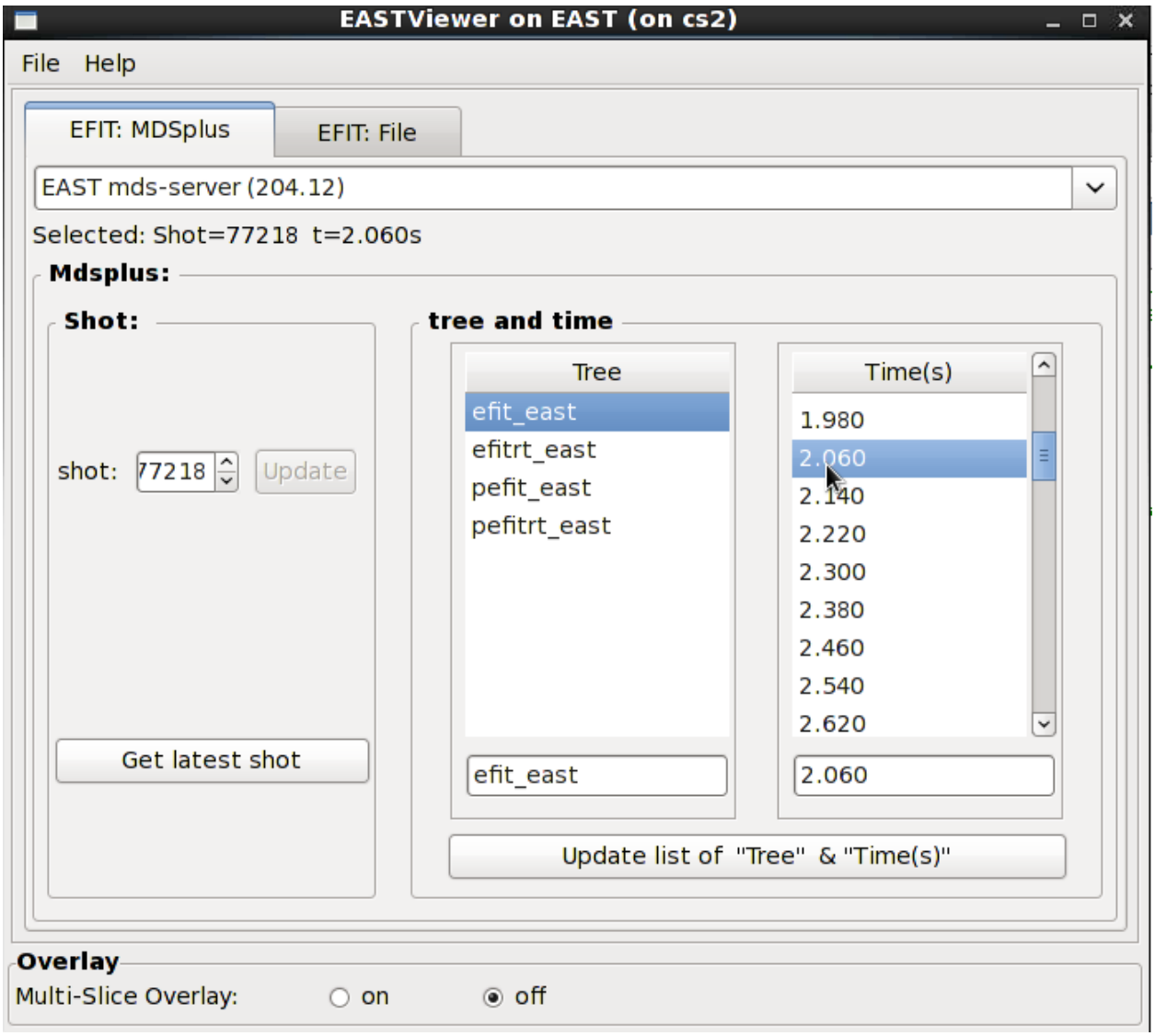


Figure 4.10: Using `eastviewer` to compare the simulation shape and parameters with the reconstructed shape and parameters

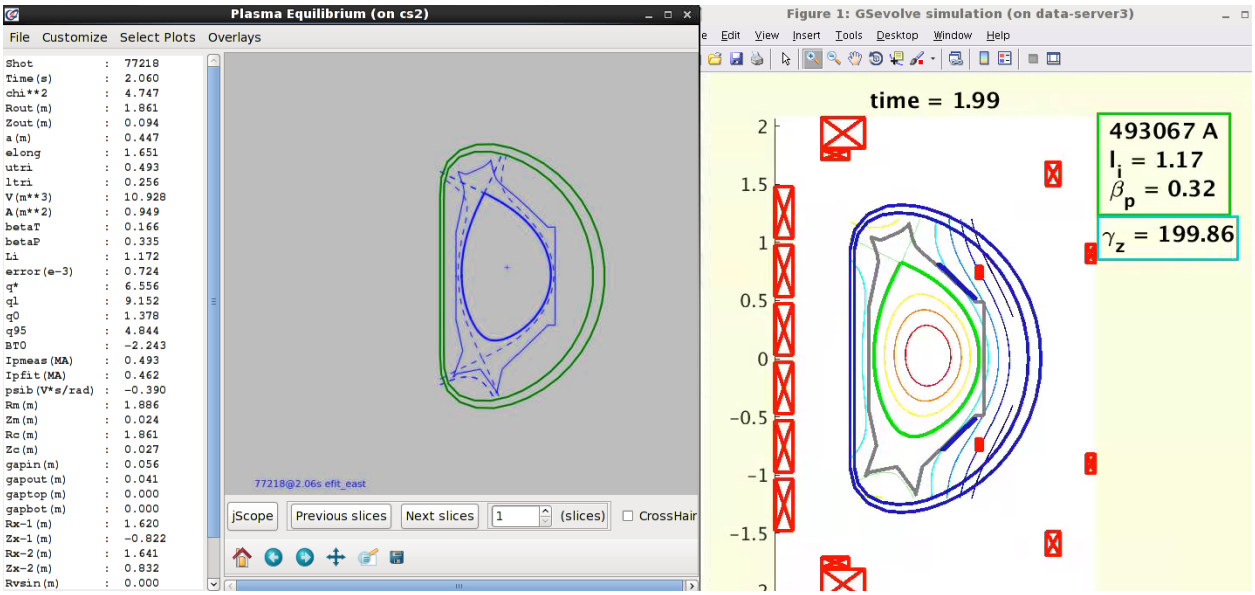


Figure 4.11: A comparison between the *Plasma Equilibrium* window and the *GSevolve simulation* window.

Chapter 5

Conclusion and Discussion

5.1 Summary

This thesis presents a unique and novel configuration for the EAST machine; demonstrating that not only is a negative-triangularity configuration possible for EAST but by extension, it shows that any D-shaped, fully superconducting tokamak should be able to recreate the NTC while keeping the plasma safely diverted on to the target plates. This thesis presents several firsts for EAST: (1) the first application of the NTC at EAST; (2) the first successful discharge of a configuration designed completely by GSdesign; (3) the first plasma response simulation for a full discharge from EAST.

Chapter 1 presented an introduction to nuclear fusion, the triple-product, magnetic confinement, tokamak physics, and negative-triangularity. The motivation for this work is to understand better the underlying physics of negative-triangularity and its apparent stability advantages and reduced heat-load.

Chapter 2 introduced EAST and compared its geometry and characteristics with those of DIII-D, TCV, and ITER. The design procedure using GSdesign, `Matlab`, and TOKSYS was presented in detail. The optimization for the NTC was also described and the final design presented with all of its considerations.

Chapter 3 presented a comparison between the experimental results and the final design from

Chapter 2. While the design results were better than expected in the final discharge, the limitations of the EAST machine were made apparent and a brief explanation of these limitations was also discussed. Notwithstanding, the final results are very encouraging and future NTC experiments are scheduled.

Chapter 4 described in great detail the initial simulations of the plasma response model for a full discharge at EAST. The success of this model in combination with the success of the NTC discharge has provided ample encouragement for the future work described in the final section.

5.2 Ongoing and Future Work

As of the time of writing, the author is in collaboration with scientists from DIII-D to refine the plasma response model (`east.slx`) such that the NTC can be successfully simulated. However, to complete this work, more data on NTC discharges will be required. Unfortunately, with the upgrade of EAST underway, there will be no new data until 2021 at the earliest. Nevertheless, several more NTC designs (a range from $-0.4 \leq \delta_L \leq 0.4$) have already been made and are waiting for EAST to be back online. In the meantime, the plasma response model can be expanded and refined for a variety of discharges, not limited to the NTC. This will eventually include auxiliary heating, more advanced diagnostics, disruption suppression through sonic beam injection, and others. While these additions will continue to improve the model, refining of the myriad variables within the model is separate and significant task altogether.

For some of the variables, better measuring techniques such as those suggested in Section 3.2. Other techniques can also be used during the upgrade to better measure certain aspects of the tokamak directly. However, for many of the variables in the model, these types of direct and indirect measuring techniques will not suffice as they cannot be done during discharges and the values of these variables change depending upon the discharge. Currently, each time the model is expanded to include another feature, several dozen discharges must be painstakingly run through the model and so that the results of the simulation can be compared with the experimental results.

This would essentially mean a poor unfortunate graduate student would be repeating the steps in Chapter 4 hundreds of times. And with each repetition guessing at adjustments in the model and then trying again. This type of trial-and-error method is tedious, prone to human error, and highly inefficient overall.

To deal with this, the author has begun working on automating the trial-and-error method. While this is still not the most efficient method, computationally speaking, it will alleviate the tediousness and human error issues. To do this, some limitations will need to be implemented. For example, for the simulation to be useful to the average EAST operator, the simulation needs to accurately predict if a specific configuration will reach break-down and flat-top. That is to say, the plasma will instantiate and reach a controllable state. For this, the simulation would only need to simulate the first two or three seconds, not the entire discharge. Other limitations may also be implemented to reduce the computations such as only including certain diagnostics rather than all available diagnostics. With these limitations in place, several dozen discharges that all use the same diagnostics, have similar configurations, and similar duration can be "batch processed" through the simulation. This alone would save hundreds of man-hours. Then the discharges that whose simulation did not reach breakdown and flat-top can be examined individually and the variables in the model can be tweaked.

While this method would save countless hours, the tweaking of the model variables still essentially means a human would need to build an intuition for how to tweak each variable. Thus, the next step of the work would be to find a correlation between how those variables were tweaked and the parameters in the discharge configuration. For example, an obvious correlation would be the resistivity between passive structure elements and the targeted plasma current. A different plasma current would cause more or fewer eddy currents in the passive structure and more or less resistivity between them. Plotting the plasma current vs. the resistivity of certain passive structures would likely reveal some pattern.

Unfortunately, this is still a very tedious process as there are hundreds of variables in the model and hundreds of parameters for each discharge configuration. To go through and examine the

correlations of each would be a herculean task... for a human. This is where the future work, not just for this thesis, but for many tokamaks is heading: Artificial Intelligence.

Because AI is not at all the topic of this thesis, only a few sentences will be given here. Only to say that other devices have successfully implemented machine learning into plasma control and disruption mitigation systems [29], [30]. The authors successfully gave the machine learning algorithms the initial parameters and disruption results of hundreds of discharges as the "teaching" data. The machine learning algorithms were then able to find the correlations between the initial parameters of the discharge and the output from the diagnostics just before the disruption to predict the instability in real-time effectively. It seems reasonable that with similar algorithms, the variables of the plasma response model could be more accurately fit, thus making the simulated plasma highly accurate.

List of References

[1] I. E. Agency. (2019). "Electricity final consumption, world," [Online]. Available: https://www.iea.org/data-and-statistics.

[2] D. Clayton, *Principles of Stellar Evolution and Nucleosynthesis*, ser. Astronomy / Astrophysics. University of Chicago Press, 1983, ISBN: 9780226109534. [Online]. Available: https://books.google.co.kr/books?id=8HSGFThnbvkC.

[3] L. Lucas, "Comprehensive review and critical evaluation of the half-life of tritium," *Journal of Research of the National Institute of Standards and Technology*, vol. 105, Jul. 2000. DOI: 10.6028/jres.105.043.

[4] M. Kikuchi, "A review of fusion and tokamak research towards steady-state operation: A jaea contribution," *Energies*, vol. 3, Nov. 2010. DOI: 10.3390/en3111741.

[5] F. Habashi, "Plutonium, physical and chemical properties," in *Encyclopedia of Metalloproteins*, R. H. Kretsinger, V. N. Uversky, and E. A. Permyakov, Eds. New York, NY: Springer New York, 2013, pp. 1751–1753, ISBN: 978-1-4614-1533-6. DOI: 10.1007/978-1-4614-1533-6_403. [Online]. Available: https://doi.org/10.1007/978-1-4614-1533-6_403.

[6] J. Wesson and D. Campbell, *Tokamaks*, ser. International series of monographs on physics. Clarendon Press, 2004, ISBN: 9780198509226. [Online]. Available: https://books.google.co.kr/books?id=iPlAwZI6HIYC.

[7] D. Clery, Ed., *Is Lockheeds fusion project breaking new ground?* Oct. 27, 2014.

[8] F. F. Chen, *Introduction to Plasma Physics and Controlled Fusion*. Springer US, Dec. 1, 2010, 440 pp., ISBN: 1441932011. [Online]. Available: https://www.ebook.de/de/product/13413995/francis_f_chen_introduction_to_plasma_physics_and_controlled_fusion.html.

[9] J. Proll, "Trapped-particle instabilities in quasi-isodynamic stellarators," Ph.D. dissertation, Jan. 2014.

[10] R. Schneider, *Plasma Edge Physics for Tokamaks*. IPP, 2001. [Online]. Available: https://books.google.com.hk/books?id=uVIOtwAACAAJ.

[11] S. L. Chen, F. Villone, B. J. Xiao, L. Barbato, Z. P. Luo, L. Liu, S. Mastrostefano, and Z. Xing, "3D passive stabilization of n = 0 MHD modes in EAST tokamak," *Scientific Reports*, vol. 6, 32440, p. 32 440, Sep. 2016. DOI: 10.1038/srep32440.

[12] A. Cardinali, C. Castaldo, R. Cesario, L. Amicucci, A. Galli, F. Napoli, L. Panaccione, C. Riccardi, F. Santini, G. Schettini, and A. A. Tuccillo, "Radio-frequency current drive for thermonuclear fusion reactors," *Scientific Reports*, vol. 8, no. 1, p. 10 318, 2018, ISSN: 2045-2322. DOI: 10.1038/s41598-018-27996-9. [Online]. Available: https://doi.org/10.1038/s41598-018-27996-9.

[13] T. C. Luce, "An analytic functional form for characterization and generation of axisymmetric plasma boundaries," *Plasma Physics and Controlled Fusion*, vol. 55, no. 9, p. 095 009, 2013. DOI: 10.1088/0741-3335/55/9/095009. [Online]. Available: https://doi.org/10.1088%2F0741-3335%2F55%2F9%2F095009.

[14] M Kikuchi, S Medvedev, T Takizuka, A Fasoli, Y Wu, P Diamond, X Duan, Y Kishimoto, K Hanada, L Villard, O Sauter, S Coda, B Duval, H Reimerdes, S Brunner, G Merlo, J Jiang, M Wang, M Ni, D Chen, H Du, W Duan, Y Hou, A Ivanov, A Martynov, Y Poshekhonov, Y Ueda, L Yan, X Song, G Zheng, J Liu, K Nagasaki, K Imadera, K Mishra, A Fujisawa, K Nakamura, H Zushi, M. J. Pueschel, X. Z. Xu, P hu, D Told, G. Q. Li, M Furukawa, T Ozeki, K Shimizu, K Kawashima, H Urano, M Honda, T Ando, and M Kuriyama, "Perspective of negative triangularity tokamak as fusion energy system," *Europhysics Conference Abstracts*, vol. 39E, P4.179, 2015. [Online]. Available: http://infoscience.epfl.ch/record/217063.

[15] M. Lanctot, R. Buttery, J. de Grassie, T. Evans, N. Ferraro, J. Hanson, S. Haskey, R. Moyer, R. Nazikian, T. Osborne, D. Orlov, P. Snyder, M. Wade, and the DIII-D Team, "Sustained suppression of type-i edge-localized modes with dominantly nă=ă2 magnetic fields in diii-d," *Nuclear Fusion*, vol. 53, no. 8, p. 083 019, 2013. [Online]. Available: http://stacks.iop.org/0029-5515/53/i=8/a=083019.

[16] M Kikuchi, A Fasoli, T Takizuka, P. Diamond, S. Medvedev, X. Duan, H Zushi, M. Furukawa, Y Kishimoto, Y Wu, O Sauter, L. Villard, S. Brunner, G. Merlo, J. Kwon, G Zheng, K. Mishra, M Honda, H Urano, and F Sano, "Negative triangularity tokamak as fusion energy system," in *Conference Proceedings Paper - Energies Whither Energy Conversion? Present Trends, Current Problems and Realistic Future Solutions*, Mar. 2014. DOI: 10.3390/ece-1-e002. [Online]. Available: https://www.researchgate.net/publication/263441260_Negative_Triangularity_Tokamak_as_Fusion_Energy_System.

[17] R. Goldston, "Heuristic drift-based model of the power scrape-off width in low-gas-puff h-mode tokamaks," *Nuclear Fusion*, vol. 52, no. 1, p. 013 009, 2012. [Online]. Available: http://stacks.iop.org/0029-5515/52/i=1/a=013009.

[18] A. Pochelon, P. Angelino, R. Behn, S. Brunner, S. Coda, N. Kirneva, S. Y. Medvedev, H. Reimerdes, J. Rossel, O. Sauter, *et al.*, "Recent tcv results-innovative plasma shaping to improve plasma properties and insight," *Plasma and Fusion Research*, vol. 7, pp. 2 502 148–2 502 148, 2012. DOI: 10.1585/pfr.7.2502148. [Online]. Available: http://www.jspf.or.jp/PFR/PDF/pfr2012_07-2502148.pdf.

[19] J.-M. Moret, S. Franke, H. Weisen, M. Anton, R. Behn, B. P. Duval, F. Hofmann, B. Joye, Y. Martin, C. Nieswand, Z. A. Pietrzyk, and W. van Toledo, "Influence of plasma shape on transport in the tcv tokamak," *Physical Review Letters*, vol. 79, pp. 2057–2060, 11 1997. DOI: 10.1103/PhysRevLett.79.2057. [Online]. Available: https://inis.iaea.org/collection/NCLCollectionStore/_Public/28/024/28024310.pdf.

[20] S. Medvedev, A. Ivanov, A. Martynov, Y. Poshekhonov, R. Behn, Y. Martin, A. Pochelon, O. Sauter, and L. Villard, "Beta limits and edge stability for negative triangularity plasma in tcv tokamak," *ECA*, vol. 32D, P–1.072, 2008. [Online]. Available: http://epsppd.epfl.ch/Hersonissos/pdf/P1_072.pdf.

[21] S. Medvedev, A. Ivanov, A. Martynov, Y. Poshekhonov, M. Kikuchi, A. Pochelon, H. Reimerdes, O. Sauter, and L. Villard, "Stability limits for tokamak plasma with negative triangularity," *Europhysics Conference Abstracts*, vol. 38F, no. P4.039, 2014. [Online]. Available: http://ocs.ciemat.es/EPS2014PAP/pdf/P4.039.pdf.

[22] S. Medvedev, M. Kikuchi, L. Villard, T. Takizuka, P. Diamond, H. Zushi, K. Nagasaki, X. Duan, Y. Wu, A. Ivanov, A. Martynov, Y. Poshekhonov, A. Fasoli, and O. Sauter, "The negative triangularity tokamak: Stability limits and prospects as a fusion energy system," *Nuclear Fusion*, vol. 55, no. 6, p. 063 013, 2015. DOI: 10.1088/0029-5515/55/6/063013. [Online]. Available: http://stacks.iop.org/0029-5515/55/i=6/a=063013.

[23] S. Y. Medvedev, M Kikuchi, T Takizuka, A. Ivanov, A. Martynov, Y. Y. Poshekhonov, A Merle, O Sauter, L Villard, D Chen, *et al.*, "Single null divertor in negative triangularity tokamak," *FEC*, 2016. [Online]. Available: https://conferences.iaea.org/indico/event/98/session/21/contribution/3.pdf.

[24] M. E. Austin, A. Marinoni, M. L. Walker, M. W. Brookman, J. S. deGrassie, A. W. Hyatt, G. R. McKee, C. C. Petty, T. L. Rhodes, S. P. Smith, C. Sung, K. E. Thome, and A. D. Turnbull, "Achievement of reactor-relevant performance in negative triangularity shape in the diii-d tokamak," *Phys. Rev. Lett.*, vol. 122, p. 115 001, 11 2019. DOI: 10.1103/PhysRevLett.122.115001. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.122.115001.

[25] B. Xiao, Q. Yuan, D. Humphreys, M. Walker, A. Hyatt, J. Leuer, G. Jackson, D. Mueller, B. Penaflor, D. Pigrowski, R. Johnson, A. Welander, R. Zhang, Z. Luo, Y. Guo, Z. Xing, and Y. Zhang, "Recent plasma control progress on east," *Fusion Engineering and Design*, vol. 87, no. 12, pp. 1887 –1890, 2012, Proceedings of the 8th IAEA Technical Meeting on Control, Data Acquisition, and Remote Participation for Fusion Research, ISSN: 0920-3796. DOI: https://doi.org/10.1016/j.fusengdes.2012.06.013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S092037961200316X.

[26] M Mateev, E Benova, S. Medvedev, A. Ivanov, A. Martynov, Y. Yu. Poshekhonov, Y. Martin, J. Moret, F Piras, A. Pochelon, O Sauter, and L. Villard, "Stability of snowflake diverted and negative triangularity plasmas in the tcv tokamak," *36th EPS Conference on Plasma Physics 2009, EPS 2009 - Europhysics Conference Abstracts*, vol. 33, Jan. 2009.

[27] H. Anand, S. Coda, F. Felici, C. Galperti, and J.-M. Moret, "A novel plasma position and shape controller for advanced configuration development on the tcv tokamak," *Nuclear Fusion*, vol. 57, no. 12, p. 126 026, 2017. [Online]. Available: http://stacks.iop.org/0029-5515/57/i=12/a=126026.

[28] Q. Yuan, B. Xiao, Z. Luo, M. Walker, A. Welander, A. Hyatt, J. Qian, R. Zhang, D. Humphreys, J. Leuer, R. Johnson, B. Penaflor, and D. Mueller, "Plasma current, position and shape feedback control on EAST," *Nuclear Fusion*, vol. 53, no. 4, p. 043 009, 2013. DOI:

10.1088/0029-5515/53/4/043009. [Online]. Available: https://doi.org/10.1088%2F0029-5515%2F53%2F4%2F043009.

[29] A. Pau, A. Fanni, B. Cannas, S. Carcangiu, G Pisano, G Sias, P Sparapani, M Baruzzo, A. Murari, F Rimini, M. Tsalas, P. De Vries, and J. Contributors, "A first analysis of jet plasma profile-based indicators for disruption prediction and avoidance," *IEEE Transactions on Plasma Science*, vol. PP, Jun. 2018. DOI: 10.1109/TPS.2018.2841394.

[30] Y. Fu, D. Eldon, K. Erickson, K. Kleijwegt, L. Lupin-Jimenez, M. D. Boyer, N. Eidietis, N. Barbour, O. Izacard, and E. Kolemen, "Machine learning control for disruption and tearing mode avoidance," *Physics of Plasmas*, vol. 27, no. 2, p. 022 501, 2020. DOI: 10.1063/1.5125581. eprint: https://doi.org/10.1063/1.5125581. [Online]. Available: https://doi.org/10.1063/1.5125581.

APPENDICES

Appendix A

A Derivation of the Grad-Shafronov Equation

A.1 GS Derivation

This derivation refers to the *Navy Research Laboratory Plasma Formulary* abbreviated as *NRL*, in particular the vector identities. An on-line copy can be found at `https://www.nrl.navy.mil/ppd/content/nrl-plasma-formulary`.

Begin with a few assumptions about the magnetic field and magnetic flux

$$\boldsymbol{B} = \boldsymbol{B_T} + \boldsymbol{B_P} = B_\varphi \boldsymbol{e_\varphi} + \frac{1}{R}\nabla\psi \times \boldsymbol{e_\varphi}, \qquad B_z = \frac{1}{R}\frac{\partial\psi}{\partial R}, \qquad \psi_P = 2\pi\psi\ (R, z = 0) \tag{A.1}$$

Where $\boldsymbol{B_T} = B_\varphi \boldsymbol{e_\varphi}$ and $\boldsymbol{B_P} = \frac{1}{R}\nabla\psi \times \boldsymbol{e_\varphi}$. Also note that both B_φ and ψ are not functions of φ. So any derivatives with respect to φ for them will be 0. Now that the functions of the magnetic field and flux have been established, begin with $\boldsymbol{J}$ and evaluate it through Ampere's Law

$$\frac{4\pi}{c}\boldsymbol{J} = \nabla \times \boldsymbol{B} = \nabla \times \boldsymbol{B_T} + \nabla \times \boldsymbol{B_p}$$

To prove that $\nabla \times \boldsymbol{B_T} = \frac{1}{R}\nabla\left(RB_\varphi\right) \times \boldsymbol{e_\varphi}$ use the *NRL Plasma Formulary* vector identities 8 and 10.

$$\begin{aligned}
\nabla \times \boldsymbol{B_T} &= \frac{1}{R}\nabla\left(RB_\varphi\right) \times \boldsymbol{e_\varphi} \\
\nabla \times \left(B_\varphi \boldsymbol{e_\varphi}\right) &= \frac{1}{R}\left(R\nabla B_\varphi + B_\varphi \nabla R\right) \times \boldsymbol{e_\varphi} \\
B_\varphi \nabla \times \boldsymbol{e_\varphi} + \nabla B_\varphi \times \boldsymbol{e_\varphi} &= \frac{1}{R}\left(R\nabla B_\varphi + B_\varphi \nabla R\right) \times \boldsymbol{e_\varphi} \\
\frac{B_\varphi}{R}\boldsymbol{e}_z + \nabla B_\varphi \times \boldsymbol{e_\varphi} &= \nabla B_\varphi \times \boldsymbol{e_\varphi} + \frac{B_\varphi}{R}\cancel{\nabla R}^{\boldsymbol{e_R}} \times \boldsymbol{e_\varphi} \\
\frac{B_\varphi}{R}\boldsymbol{e}_z + \nabla B_\varphi \times \boldsymbol{e_\varphi} &= \nabla B_\varphi \times \boldsymbol{e_\varphi} + \frac{B_\varphi}{R}\boldsymbol{e}_z
\end{aligned}$$

Move on to the poloidal field and to show that $\nabla \times \boldsymbol{B_p} = \left(-\frac{1}{R}\nabla^2\psi + \frac{2}{R^2}\frac{\partial\psi}{\partial R}\right)\boldsymbol{e_\varphi} = -\frac{1}{R}\Delta^*\psi\boldsymbol{e_\varphi}$, use *NRL* vector identities 7 and 10.

$$
\begin{aligned}
\left(-\frac{1}{R}\nabla^2\psi + \frac{2}{R^2}\frac{\partial\psi}{\partial R}\right)\boldsymbol{e_\varphi} &= \nabla \times \boldsymbol{B_p} \\
&= \nabla \times \left(\frac{1}{R}\nabla\psi \times \boldsymbol{e_\varphi}\right) \\
&= \cancelto{0}{\frac{1}{R}\nabla\psi\left(\nabla \cdot \boldsymbol{e_\varphi}\right)} - \boldsymbol{e_\varphi}\left(\nabla \cdot \left(\frac{1}{R}\nabla\psi\right)\right) \\
&\quad + \cancelto{0}{\left(\boldsymbol{e_\varphi} \cdot \nabla\right)\left(\frac{1}{R}\nabla\psi\right)} - \left(\left(\frac{1}{R}\nabla\psi\right) \cdot \nabla\right)\boldsymbol{e_\varphi} \\
&= -\boldsymbol{e_\varphi}\left(\nabla \cdot \left(\frac{1}{R}\nabla\psi\right)\right) - \left(\left(\frac{1}{R}\nabla\psi\right) \cdot \nabla\right)\boldsymbol{e_\varphi} \\
&= -\left(\frac{1}{R}\nabla \cdot \nabla\psi + \nabla\psi \cdot \nabla\frac{1}{R}\right)\boldsymbol{e_\varphi} - \left(\left(\frac{1}{R}\nabla\psi\right) \cdot \nabla\right)\boldsymbol{e_\varphi} \\
&= -\left(\frac{1}{R}\nabla^2\psi + \nabla\psi \cdot \frac{-1}{R^2}\boldsymbol{e_R}\right)\boldsymbol{e_\varphi} - \left(\left(\frac{1}{R}\nabla\psi\right) \cdot \nabla\right)\boldsymbol{e_\varphi} \\
&= -\left(\frac{1}{R}\nabla^2\psi - \frac{\partial\psi}{\partial R}\frac{1}{R^2}\right)\boldsymbol{e_\varphi} - \left(\left(\frac{1}{R}\nabla\psi\right) \cdot \nabla\right)\boldsymbol{e_\varphi} \\
&= \left(-\frac{1}{R}\nabla^2\psi + \frac{1}{R^2}\frac{\partial\psi}{\partial R}\right)\boldsymbol{e_\varphi} - \left(\frac{1}{R}\left[\frac{\partial\psi}{\partial R}\boldsymbol{e_R} + \frac{\partial\psi}{\partial z}\boldsymbol{e_z}\right] \cdot \nabla\right)\boldsymbol{e_\varphi} \\
&= \left(-\frac{1}{R}\nabla^2\psi + \frac{1}{R^2}\frac{\partial\psi}{\partial R}\right)\boldsymbol{e_\varphi} - \frac{1}{R}\left[\frac{\partial\psi}{\partial R}\frac{1}{R}\frac{\partial\left(R\right)}{\partial R} + \frac{\partial\psi}{\partial z}\frac{\partial\left(\right)}{\partial z}\right]\boldsymbol{e_\varphi} \\
&= \left(-\frac{1}{R}\nabla^2\psi + \frac{1}{R^2}\frac{\partial\psi}{\partial R}\right)\boldsymbol{e_\varphi} - \frac{1}{R^2}\left[\frac{\partial\psi}{\partial R}\frac{\partial\left(R\boldsymbol{e_\varphi}\right)}{\partial R} + \cancelto{0}{\frac{\partial\psi}{\partial z}\frac{\partial\left(\boldsymbol{e_\varphi}\right)}{\partial z}}\right] \\
&= \left(-\frac{1}{R}\nabla^2\psi + \frac{1}{R^2}\frac{\partial\psi}{\partial R}\right)\boldsymbol{e_\varphi} + \frac{1}{R^2}\frac{\partial\psi}{\partial R}\boldsymbol{e_\varphi} \\
&= \left(-\frac{1}{R}\nabla^2\psi + \frac{2}{R^2}\frac{\partial\psi}{\partial R}\right)\boldsymbol{e_\varphi}
\end{aligned}
$$

Finally, the curl of the toroidal and poloidal magnetic fields can be summarized.

$$
\begin{aligned}
\nabla \times \boldsymbol{B_p} &= \left(-\frac{1}{R}\nabla^2\psi + \frac{2}{R^2}\frac{\partial\psi}{\partial R}\right)\boldsymbol{e_\varphi} = -\frac{1}{R}\Delta^*\psi\boldsymbol{e_\varphi} \\
\nabla \times \boldsymbol{B_T} &= \nabla \times B_\varphi\boldsymbol{e_\varphi} = \frac{1}{R}B_\varphi\boldsymbol{e_z} + \nabla B_\varphi \times \boldsymbol{e_\varphi} = \frac{1}{R}\nabla\left(RB_\varphi\right) \times \boldsymbol{e_\varphi}
\end{aligned}
$$

Where the elliptic operator is given as

$$\Delta^* \psi = R \frac{\partial}{\partial R} \left(\frac{1}{R} \frac{\partial \psi}{\partial R} \right) + \frac{\partial^2 \psi}{\partial Z^2} \tag{A.2}$$

Thus the current density can now be written as

$$\frac{1}{c} \boldsymbol{J} = \frac{1}{4\pi R} \left(-\Delta^* \psi \boldsymbol{e_\varphi} + \nabla \left(R B_\varphi \right) \times \boldsymbol{e_\varphi} \right) \tag{A.3}$$

Now for the force balance equation:

$$\nabla p = \frac{1}{c} \boldsymbol{J} \times \boldsymbol{B} \tag{A.4}$$

It can be proven that the cross product means both $\boldsymbol{B} \cdot \nabla p = 0$ and $\boldsymbol{J} \cdot \nabla p = 0$ which implies that $\boldsymbol{B_P} \cdot \nabla p = 0$ which also implies that $\nabla p \parallel \nabla \psi$ thus, p and RB_φ are parallel to ψ and so can be defined as

$$\begin{aligned} p &= p(\psi) \\ F &\equiv R B_\varphi = F(\psi) \end{aligned}$$

It can also be shown that $I_P = 2\pi F(\psi)$. Let $\nabla p = \frac{\mathrm{d}p}{\mathrm{d}\psi} \nabla \psi$, and $\nabla F = \frac{\mathrm{d}F}{\mathrm{d}\psi} \nabla \psi$. Substitute Equation (A.1) and Equation (A.3) into Equation (A.4) then dotting each side with $\nabla \psi$, assuming $\nabla \psi \cdot \nabla \psi = \left| \nabla \psi \right|^2$, then work out the final form of the equation:

$$\nabla p \cdot \nabla \psi = \left[\frac{1}{4\pi R} \left(-\Delta^* \psi \boldsymbol{e_\varphi} + \nabla F \times \boldsymbol{e_\varphi} \right) \times \left(B_\varphi \boldsymbol{e_\varphi} + \frac{1}{R} \nabla \psi \times \boldsymbol{e_\varphi} \right) \right] \cdot \nabla \psi$$

$$\left(\left(\frac{\mathrm{d}p}{\mathrm{d}\psi} \right) \nabla \psi \right) \cdot \nabla \psi = \frac{1}{4\pi R} \left[\cancelto{0}{\left(-\Delta^* \psi \boldsymbol{e_\varphi} \right) \times B_\varphi \boldsymbol{e_\varphi}} + \left(\nabla F \times \boldsymbol{e_\varphi} \right) \times B_\varphi \boldsymbol{e_\varphi} + \left(-\Delta^* \psi \boldsymbol{e_\varphi} \right) \times \left(\frac{1}{R} \nabla \psi \times \boldsymbol{e_\varphi} \right) \right.$$

$$\left. + \left(\nabla F \times \boldsymbol{e_\varphi} \right) \times \left(\frac{1}{R} \nabla \psi \times \boldsymbol{e_\varphi} \right) \right] \cdot \nabla \psi$$

$$\left(\frac{\mathrm{d}p}{\mathrm{d}\psi}\right)\left|\nabla\psi\right|^2 = \frac{1}{4\pi R}\left[\left(\nabla F\times\boldsymbol{e}_{\varphi}\right)\times B_{\varphi}\boldsymbol{e}_{\varphi}+\left(-\Delta^*\psi\boldsymbol{e}_{\varphi}\right)\times\left(\frac{1}{R}\nabla\psi\times\boldsymbol{e}_{\varphi}\right)+\left(\nabla F\times\boldsymbol{e}_{\varphi}\right)\right.$$
$$\left.\times\left(\frac{1}{R}\nabla\psi\times\boldsymbol{e}_{\varphi}\right)\right]\cdot\nabla\psi \quad \text{NRL VI (3), 3}^{\text{rd}}\text{ term}$$
$$= \frac{1}{4\pi R}\left[\left(\nabla F\times\boldsymbol{e}_{\varphi}\right)\times B_{\varphi}\boldsymbol{e}_{\varphi}+\left(-\Delta^*\psi\boldsymbol{e}_{\varphi}\right)\times\left(\frac{1}{R}\nabla\psi\times\boldsymbol{e}_{\varphi}\right)\right.$$
$$\left.+\cancel{\left[\left(\boldsymbol{e}_{\varphi}\times\left(\nabla F\times\boldsymbol{e}_{\varphi}\right)\right)\times\nabla\psi\right]}-\cancel{\left[\boldsymbol{e}_{\varphi}\times\left(\left(\nabla F\times\boldsymbol{e}_{\varphi}\right)\times\nabla\psi\right)\right]}\right]\cdot\nabla\psi$$
NRL VI (2)
$$= \frac{1}{4\pi R}\left[\left(\frac{\mathrm{d}F}{\mathrm{d}\psi}\nabla\psi\times\boldsymbol{e}_{\varphi}\right)\times B_{\varphi}\boldsymbol{e}_{\varphi}-\frac{1}{4\pi R}\Delta^*\psi\left(\cancel{\left(\boldsymbol{e}_{\varphi}\cdot\boldsymbol{e}_{\varphi}\right)}^{1}\nabla\psi-\cancel{\left(\boldsymbol{e}_{\varphi}\cdot\nabla\psi\right)\boldsymbol{e}_{\varphi}}^{0}\right)\right]$$
$\cdot\nabla\psi$ NRL VI (2)
$$= \frac{1}{4\pi R}\left[\frac{\mathrm{d}F}{\mathrm{d}\psi}\left(B_{\varphi}\cancel{\left(\boldsymbol{e}_{\varphi}\cdot\nabla\psi\right)}^{0}\boldsymbol{e}_{\varphi}-B_{\varphi}\cancel{\left(\boldsymbol{e}\cdot\boldsymbol{e}_{\varphi}\right)}^{1}\nabla\psi\right)-\frac{1}{4\pi R}\Delta^*\psi\nabla\psi\right]\cdot\nabla\psi$$
$$= \frac{1}{4\pi R}\left[-B_{\varphi}\frac{\mathrm{d}F}{\mathrm{d}\psi}\nabla\psi-\frac{1}{4\pi R}\Delta^*\psi\nabla\psi\right]\cdot\nabla\psi$$
$$\left(\frac{\mathrm{d}p}{\mathrm{d}\psi}\right)\left|\nabla\psi\right|^2 = \frac{1}{4\pi R}\left[-B_{\varphi}\frac{\mathrm{d}F}{\mathrm{d}\psi}\nabla\psi\cdot\nabla\psi-\frac{1}{4\pi R}\Delta^*\psi\nabla\psi\cdot\nabla\psi\right]$$
$$\left(\frac{\mathrm{d}p}{\mathrm{d}\psi}\right)\cancel{\left|\nabla\psi\right|^2} = \frac{1}{4\pi R}\left[-B_{\varphi}\frac{\mathrm{d}F}{\mathrm{d}\psi}\cancel{\left|\nabla\psi\right|^2}-\frac{1}{4\pi R}\Delta^*\psi\cancel{\left|\nabla\psi\right|^2}\right]$$
$$\frac{\mathrm{d}p}{\mathrm{d}\psi} = -\frac{B_{\varphi}}{4\pi R}\frac{\mathrm{d}F}{\mathrm{d}\psi}-\frac{1}{4\pi R^2}\Delta^*\psi$$
$$\Delta^*\psi = -4\pi R^2\frac{\mathrm{d}p}{\mathrm{d}\psi}-RB_{\varphi}\frac{\mathrm{d}F}{\mathrm{d}\psi} = -4\pi R^2\frac{\mathrm{d}p}{\mathrm{d}\psi}-F\frac{\mathrm{d}F}{\mathrm{d}\psi} \tag{A.5}$$

Including the elliptic operator, the full Grad-Shafranov equation becomes

$$R\frac{\partial}{\partial R}\left(\frac{1}{R}\frac{\partial\psi}{\partial R}\right)+\frac{\partial^2\psi}{\partial Z^2} = -4\pi R^2\frac{\mathrm{d}p}{\mathrm{d}\psi}-F\frac{\mathrm{d}F}{\mathrm{d}\psi}$$

The subsections below include an analytical solution to the GS equation for further understanding. However, it is not necessary to go any further if you have understood the above derivation and its physical meaning well.

A.1.1 0^{th} Order Solution

Begin by converting the elliptic operator given by Equation (A.2) into toroidal coordinates using

$$R = R_0 + r\cos(\vartheta), \qquad z = r\cos(\vartheta), \qquad \varphi' = -\varphi \tag{A.6}$$

$$\begin{aligned} \Delta^*\psi &= R\frac{\partial}{\partial R}\left(\frac{1}{R}\frac{\partial\psi}{\partial R}\right) + \frac{\partial^2\psi}{\partial Z^2} \\ \Rightarrow \Delta^*\psi &= \frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial\psi}{\partial r}\right) + \frac{1}{r^2}\frac{\partial^2\psi}{\partial\vartheta^2} - \frac{1}{R_0}\left(\cos(\vartheta)\frac{\partial\psi}{\partial r} - \frac{\sin(\vartheta)}{r}\frac{\partial\psi}{\partial\vartheta}\right) \end{aligned} \tag{A.7}$$

Assume that the plasma is surrounded by a perfectly conducting surface of circular cross section of radius $r = a$. Then the boundary condition is

$$\psi_{r=a} = \psi(a,\vartheta) = const, \qquad \psi_{r<a} = \psi(r,\vartheta) \neq const$$

Use the low-β assumption $\beta_P \sim 1$ for tokamaks which means

$$\frac{B_P}{B_T} \sim \epsilon \Rightarrow \beta_P = \frac{8\pi\langle p\rangle}{B_P^2}, \quad \beta_T = \frac{8\pi\langle p\rangle}{B_T^2} \Rightarrow \beta_P = \frac{B_T^2}{B_P^2}\beta_T = \frac{1}{\epsilon^2}\beta_T \Rightarrow \beta_T \sim \epsilon^2$$

This also implies that the magnetic winding index is on the order of unity, $q \sim 1$.

Furthermore, assume that the solution for the flux surface will be of the form

$$\psi(r,\vartheta) = \psi_0(r) + \psi_1(r,\vartheta) + \cancel{\psi_2(r,\vartheta) + \ldots}^{\sim\epsilon^2}$$

Here, $\psi_0(r)$ is the 0^{th} order solution where the tokamak aspect ratio goes to zero, $\frac{r}{R_0} \to 0 \Rightarrow \epsilon \to 0$. Assume the 0^{th} order terms to be ~ 1. In this limit the poloidal magnetic field is only a function of r meaning $B_P(r,\vartheta) \to B_{P1}(r)$. Use the subscript "1" here because it is a 1^{st} order quantity. And the relationship between B_{P1} and $\psi_0(r)$ can be expressed as

$$B_{P1}(r) = \frac{1}{R_0}\frac{\mathrm{d}\psi_0}{\mathrm{d}r}$$

The toroidal magnetic field in the 0^{th} limit is a constant $B_T(r,\vartheta) = B_{T0} = const.$ Estimate $\psi \sim rR_0B_P$, then Equation (A.7) and divide by B_{T0} to make the equations dimensionless and thus easier to analyze the order of each term.

$$
\begin{aligned}
\Delta^*\psi &= \frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial\psi}{\partial r}\right) + \frac{1}{r^2}\frac{\partial^2\psi}{\partial\vartheta^2} - \frac{1}{R_0}\left(\cos(\vartheta)\frac{\partial\psi}{\partial r} - \frac{\sin(\vartheta)}{r}\frac{\partial\psi}{\partial\vartheta}\right)\\
\frac{1}{B_{T0}}\Delta^*\psi &= \frac{1}{B_{T0}}\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial\psi}{\partial r}\right) + \frac{1}{B_{T0}}\frac{1}{r^2}\frac{\partial^2\psi}{\partial\vartheta^2} - \frac{1}{B_{T0}}\frac{1}{R_0}\left(\cos(\vartheta)\frac{\partial\psi}{\partial r} - \frac{\sin(\vartheta)}{r}\frac{\partial\psi}{\partial\vartheta}\right)\\
&= \frac{1}{B_{T0}}\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial\psi}{\partial r}\right) + \frac{1}{B_{T0}}\frac{1}{r^2}\cancelto{0}{\frac{\partial^2 rR_0B_P}{\partial\vartheta^2}} - \frac{1}{B_{T0}}\frac{1}{R_0}\left(\cos(\vartheta)\frac{\partial rR_0B_P}{\partial r} - \frac{\sin(\vartheta)}{r}\cancelto{0}{\frac{\partial rR_0B_P}{\partial\vartheta}}\right)\\
&= \frac{1}{B_{T0}}\frac{1}{r}\frac{\partial}{\partial r}\left(rR_0B_P\right) - \frac{1}{B_{T0}}\frac{1}{\cancel{R_0}}\left(\cos(\vartheta)\frac{\cancel{\partial r}\cancel{R_0}B_P}{\cancel{\partial r}}\right)\\
&= \frac{1}{B_{T0}}\frac{R_0B_P}{r} - \cancelto{\sim\epsilon}{\frac{B_P}{B_{T0}}\cos(\vartheta)}\left(\frac{1}{B_{T0}}\Delta^*\psi\right)_{0^{th}}\\
&= \frac{R_0B_P}{rB_{T0}}\\
&\sim 1
\end{aligned}
$$

Before applying this same procedure to the RHS of Equation (A.5), linearize the derivatives of p and F with respect to ψ. So, let

$$
\begin{aligned}
p &= \frac{\psi^2}{\psi_{ma}^2}p_{ma} & F^2 &= R_0^2B_{T0}^2\left(1 + \frac{\psi^2}{\psi_{ma}^2}b_{ma}\right)\\
\Rightarrow \frac{\mathrm{d}p}{\mathrm{d}\psi} &= 2\frac{\psi}{\psi_{ma}^2}p_{ma} & \Rightarrow \frac{\mathrm{d}F^2}{2\,\mathrm{d}\psi} &= R_0^2B_{T0}^2\frac{\psi}{\psi_{ma}^2}b_{ma}
\end{aligned}
\tag{A.8}
$$

where p_{ma}, ψ_{ma}, and b_{ma} are the pressure, poloidal flux, and diamagnetic term at the magnetic axis, respectively. Use these to get the order of ϵ of the individual terms. Note that if $b_{ma} < 0$ the term is diamagnetic and if $b_{ma} > 0$ the term paramagnetic and that $b_{ma} \sim \epsilon^2$. Finally, assume $\psi \sim rR_0B_P$ and the same for ψ_{ma}.

Substitute, simplify, and reorganize the RHS of Equation (A.5).

$$\begin{aligned}
\Delta^*\psi &= -4\pi R^2 \frac{\mathrm{d}p}{\mathrm{d}\psi} - F\frac{\mathrm{d}F}{\mathrm{d}\psi} \\
&= -4\pi \left(R_0 + r\cos(\vartheta)\right)^2 2\frac{\psi}{\psi_{ma}^2} p_{ma} - \frac{\mathrm{d}F^2}{2\,\mathrm{d}\psi} \\
&= -4\pi \left(R_0^2 + \cancelto{\sim \epsilon}{R_0 r\cos(\vartheta)} + \cancelto{\sim \epsilon^2}{\cos(\vartheta)^2} \right) 2\frac{\psi}{\psi_{ma}^2} p_{ma} - R_0^2 B_{T0}^2 \frac{\psi}{\psi_{ma}^2} b_{ma} \\
&= -\frac{R_0^2 B_{T0}^2}{\psi_{ma}^2} \left(\frac{8\pi p_{ma}}{B_{T0}^2} + b_{ma} \right) \psi \\
&= -\frac{R_0^2 B_{T0}^2}{\psi_{ma}^2} \left(\beta_{Tma} + b_{ma} \right) \psi \\
\frac{1}{B_{T0}}\Delta^*\psi &= -\frac{1}{\cancel{B_{T0}}} \frac{\cancel{R_0^2} \cancel{B_{T0}^2}}{r\cancel{r}\cancel{R^2} \cancel{B_P^2}} \left(\beta_{Tma} + b_{ma} \right) \cancel{r} R \cancel{B_P} \\
&= -\frac{R_0 B_{T0}}{r B_P} \left(\beta_{Tma} + b_{ma} \right) \\
\frac{1}{B_{T0}}\Delta^*\psi &= -\underbrace{\frac{R_0 B_{T0}}{r B_P}}_{\sim\epsilon^{-2}} \left(\underbrace{\beta_{Tma}}_{\sim\epsilon^2} + \underbrace{b_{ma}}_{\sim\epsilon^2} \right)
\end{aligned}$$

Both terms are 0^{th} order. Combine them with the 0^{th} order terms of the RHS of Equation (A.7) and let $\psi(r,\vartheta) \approx \psi_0(r)$. To make the notation more concise, let the entire quotient preceding the parenthesis on the RHS be A. This coefficient has the dimensions of the reciprocal of area.

$$\begin{aligned}
\frac{1}{r}\frac{\mathrm{d}}{\mathrm{d}r}\left(r\frac{\mathrm{d}\psi_0}{\mathrm{d}r} \right) &= -\frac{R_0^2 B_{T0}^2}{\psi_{ma}^2} \left(\beta_{Tma} + b_{ma} \right) \psi_0 \frac{1}{r}\frac{\mathrm{d}}{\mathrm{d}r}\left(r\frac{\mathrm{d}\psi_0}{\mathrm{d}r} \right) \\
\frac{1}{r}\frac{\mathrm{d}}{\mathrm{d}r}\left(r\frac{\mathrm{d}\psi_0}{\mathrm{d}r} \right) &= -A \left(\beta_{Tma} + b_{ma} \right) \psi_0 \qquad \text{(A.9)}
\end{aligned}$$

While this equation looks quite tractable, it isn't trivial (at least not for the beginner). With use of the symbolic mathematics package `Maple` as well as the "Handbook of Mathematical functions" by Abramowitz & Stegun, the solution to this was determined to be a Bessel function. One more change is made so that this equation matches the form in the "Handbook" and that is to combine A with the sum in the parentheses to form $\tilde{A}$. Finally, manipulate the equation into the form associated

with the Bessel function.

$$\frac{1}{r}\frac{\mathrm{d}}{\mathrm{d}r}\left(r\frac{\mathrm{d}\psi_0}{\mathrm{d}r}\right) = -\tilde{A}\psi_0$$
$$0 = \frac{1}{r}\frac{\mathrm{d}\psi_0}{\mathrm{d}r} + \frac{\mathrm{d}^2\psi_0}{\mathrm{d}r^2} + \tilde{A}\psi_0$$
$$= r\frac{\mathrm{d}\psi_0}{\mathrm{d}r} + r^2\frac{\mathrm{d}^2\psi_0}{\mathrm{d}r^2} + r^2\tilde{A}\psi_0$$

The general form of the Bessel function has a solution that is given by a linear combination of the two linearly independent functions $J_\nu(z)$ and $Y_\nu(z)$, the first and second kind, respectively. However, $Y_\nu(z)$ is divergent at $z = 0$ and so its coefficient of integration is set to 0. This leaves us with the possible forms of the solution for the first kind as given by Abramowitz & Stegun. Furthermore, the coefficient C_1 is obviously the value of $\psi_0(r = 0)$. Since this is the unperturbed solution, $r = 0$ corresponds to the magnetic axis as the Shafronov shift is in the perturbed solution. Which means that $C_1 = \psi_{ma}$. I will ignore this for now and just focus on the form of the solution first.

$$0 = z\frac{\mathrm{d}w}{\mathrm{d}z} + z^2\frac{\mathrm{d}^2w}{\mathrm{d}z^2} + \left(z^2 - \nu^2\right)w$$
$$w(z) = J_\nu(z)C_1 + \cancelto{divergent}{Y_\nu(z)C_2} \quad \text{full generic solution}$$
$$w(z) = J_\nu(z) = \left(\frac{z}{2}\right)^\nu \sum_{k=0}^{\infty} \frac{\left(-\frac{z^2}{4}\right)^k}{k!\,\Gamma(\nu + k + 1)} \quad \text{Ascending Series}$$
$$w(z) = J_\nu(z) = \frac{\left(\frac{z}{2}\right)^\nu}{\sqrt{\pi}\Gamma(\nu + \frac{1}{2})}\int_0^\pi \cos\left(z\cos(\theta)\right)\sin(\theta)^{2\nu}\,\mathrm{d}\theta \quad \text{Integral Representation}$$

Only the integral representation will be used and, in this case, $\nu = 0$ allowing the solution to simplify. Furthermore, $z = r\sqrt{\tilde{A}}$ and C_1 is replaced with ψ_{ma}.

$$J_0(z) = \frac{1}{\pi}\int_0^\pi \cos\left(z\cos(\theta)\right)\mathrm{d}\theta$$
$$\psi_0(r) = \psi_{ma}J_0\left(r\sqrt{\tilde{A}}\right) = \frac{\psi_{ma}}{\pi}\int_0^\pi \cos\left(r\sqrt{\tilde{A}}\cos(\theta)\right)\mathrm{d}\theta \tag{A.10}$$

A.1.2 1st Order Solution

Begin with substituting $\psi(r,\vartheta) = \psi_0(r) + \psi_1(r,\vartheta)$ into Equation (A.7) so that the 0^{th} order terms can be separated out, which were just found, and keep the 1^{st} order terms:

$$\begin{aligned}\left(\Delta^*\psi\right)_{1^{st}} = &\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial\left(\psi_0(r)+\psi_1(r,\vartheta)\right)}{\partial r}\right) + \frac{1}{r^2}\frac{\partial^2\left(\psi_0(r)+\psi_1(r,\vartheta)\right)}{\partial\vartheta^2}\\ &- \frac{1}{R_0}\left(\cos(\vartheta)\frac{\partial\left(\psi_0(r)+\psi_1(r,\vartheta)\right)}{\partial r} - \frac{\sin(\vartheta)}{r}\frac{\partial\left(\psi_0(r)+\psi_1(r,\vartheta)\right)}{\partial\vartheta}\right)\end{aligned}$$

By putting ψ_1 into the last term with the $\frac{1}{R_0}$ coefficient, this whole term becomes $\sim \epsilon^2$ because $\frac{1}{R_0} \sim \epsilon$. Instead, let $\psi = \psi_0(r)$ and note that $\frac{\mathrm{d}\psi}{\mathrm{d}\vartheta} = 0$, and finally separate out all the 0^{th} order terms in Equation (A.9).

$$\begin{aligned}\left(\Delta^*\psi\right)_{1^{st}} &= \frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial\left(\psi_0(r)+\psi_1(r,\vartheta)\right)}{\partial r}\right) + \frac{1}{r^2}\frac{\partial^2\left(\psi_0(r)+\psi_1(r,\vartheta)\right)}{\partial\vartheta^2}\\ &\quad - \frac{1}{R_0}\left(\cos(\vartheta)\frac{\partial\psi_0(r)}{\partial r} - \cancelto{0}{\frac{\sin(\vartheta)}{r}\frac{\partial\psi_0(r)}{\partial\vartheta}}\right)\\ &= \frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial\left(\psi_0(r)+\psi_1(r,\vartheta)\right)}{\partial r}\right) + \frac{1}{r^2}\frac{\partial^2\left(\psi_0(r)+\psi_1(r,\vartheta)\right)}{\partial\vartheta^2} - \frac{1}{R_0}\cos(\vartheta)\frac{\partial\psi_0(r)}{\partial r}\\ &= \frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial\psi_1(r,\vartheta)}{\partial r}\right) + \frac{1}{r^2}\frac{\partial^2\psi_1(r,\vartheta)}{\partial\vartheta^2} - \frac{1}{R_0}\cos(\vartheta)\frac{\partial\psi_0(r)}{\partial r}\end{aligned} \tag{A.11}$$

Let $\frac{\mathrm{d}p}{\mathrm{d}\psi} = \frac{\mathrm{d}p}{\mathrm{d}\psi_0} + \frac{\mathrm{d}^2p}{\mathrm{d}\psi_0^2}\psi_1 + \ldots$ and ignore all non-1^{st} order terms for this part of the analysis. Convert coordinate systems from cylindrical to toroidal with Equation (A.6). Then substitute and expand

$$\begin{aligned}\left(-4\pi R^2\frac{\mathrm{d}p}{\mathrm{d}\psi}\right)_{1^{st}} &= -4\pi\left(R_0 + r\cos(\vartheta)\right)^2\left(\frac{\mathrm{d}p}{\mathrm{d}\psi_0} + \frac{\mathrm{d}^2p}{\mathrm{d}\psi_0^2}\psi_1\right)\\ &= -4\pi\left(\cancel{R_0^2\frac{\mathrm{d}p}{\mathrm{d}\psi_0}} + 2R_0r\cos(\vartheta)\frac{\mathrm{d}p}{\mathrm{d}\psi_0} + \cancel{r^2\cos^2(\vartheta)\frac{\mathrm{d}p}{\mathrm{d}\psi_0}} + R_0^2\frac{\mathrm{d}^2p}{\mathrm{d}\psi_0^2}\psi_1\right.\\ &\qquad\left. + \cancel{2R_0r\cos(\vartheta)\frac{\mathrm{d}^2p}{\mathrm{d}\psi_0^2}\psi_1} + \cancel{r^2\cos^2(\vartheta)\frac{\mathrm{d}^2p}{\mathrm{d}\psi_0^2}\psi_1}\right)\\ &= -4\pi\left(2R_0r\cos(\vartheta)\frac{\mathrm{d}p}{\mathrm{d}\psi_0} + R_0^2\frac{\mathrm{d}^2p}{\mathrm{d}\psi_0^2}\psi_1\right)\\ &= -8\pi R_0r\frac{\mathrm{d}p}{\mathrm{d}\psi_0}\cos(\vartheta) - 4\pi R_0^2\frac{\mathrm{d}^2p}{\mathrm{d}\psi_0^2}\psi_1\end{aligned} \tag{A.12}$$

In a similar way, the last term in the RHS of Equation (A.5) also produces one 1^{st} order term after ignoring the 0^{th} order term.

$$\begin{aligned}\left(-F\frac{\mathrm{d}F}{\mathrm{d}\psi}\right)_{1^{st}} &= \cancel{-F\frac{\mathrm{d}F}{\mathrm{d}\psi_0}} - \frac{\mathrm{d}}{\mathrm{d}\psi_0}\left(F\frac{\mathrm{d}F}{\mathrm{d}\psi_0}\right)\psi_1 \\ &= -\frac{\mathrm{d}}{\mathrm{d}\psi_0}\left(\frac{\mathrm{d}F^2}{\mathrm{d}\psi_0}\right)\psi_1 \\ &= -\frac{1}{2}\frac{\mathrm{d}^2F^2}{\mathrm{d}\psi_0^2}\psi_1\end{aligned} \tag{A.13}$$

Combine Equation (A.12) and Equation (A.13), then apply Equation (A.8) for p and F to get:

$$\begin{aligned}\left(\Delta^*\psi\right)_{1^{st}} &= -8\pi R_0 r\frac{\mathrm{d}p}{\psi_0}\cos(\vartheta) - 4\pi R_0^2\frac{\mathrm{d}^2p}{\mathrm{d}\psi_0^2}\psi_1 - \frac{1}{2}\frac{\mathrm{d}^2F^2}{\mathrm{d}\psi_0^2}\psi_1 \\ &= -16\pi R_0 r\frac{\psi_0}{\psi_{ma}^2}\cos(\vartheta) - 8\pi R_0^2\frac{1}{\psi_{ma}^2}\psi_1 - R_0^2B_{T0}^2b_{ma}\frac{1}{\psi_{ma}^2}\psi_1 \\ &= -\frac{R_0^2B_{T0}^2}{\psi_{ma}^2}\left(16\pi r\frac{\psi_0}{R_0B_{T0}^2\psi_{ma}^2}p_{ma}\cos(\vartheta) + 8\pi\frac{\psi_1}{B_{T0}^2\psi_{ma}^2}p_{ma} + b_{ma}\psi_1\right) \\ &= -\frac{R_0^2B_{T0}^2}{\psi_{ma}^2}\left(\psi_1\beta_{Tma} + \frac{2r\cos(\vartheta)\psi_0}{R_0}\beta_{Tma} + b_{ma}\psi_1\right) \\ &= -A\left(\psi_1\beta_{Tma} + \frac{2r\cos(\vartheta)\psi_0}{R_0}\beta_{Tma} + b_{ma}\psi_1\right) \\ &= -\tilde{A}\psi_1 - 2A\frac{r\cos(\vartheta)\beta_{Tma}}{R_0}\psi_0\end{aligned} \tag{A.14}$$

Now put Equation (A.11) and Equation (A.14) together to form (dropping the $\psi_1(r,\vartheta)$ and $\psi_0(r)$ notation for compactness)

$$\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial\psi_1}{\partial r}\right) + \frac{1}{r^2}\frac{\partial^2\psi_1}{\partial\vartheta^2} - \frac{1}{R_0}\cos(\vartheta)\frac{\partial\psi_0}{\partial r} = -\tilde{A}\psi_1 - 2A\frac{r\cos(\vartheta)\beta_{Tma}}{R_0}\psi_0$$

Assume a circular boundary where $\psi_1(r,\vartheta) = \psi_1(r)\cos(\vartheta)$, then cancel the $\cos(\vartheta)$ from each term

$$\begin{aligned}\frac{\cancel{\cos(\vartheta)}}{r}\frac{\partial}{\partial r}\left(r\frac{\partial\psi_1}{\partial r}\right) - \frac{\psi_1}{r^2}\cancel{\cos(\vartheta)} - \frac{1}{R_0}\cancel{\cos(\vartheta)}\frac{\partial\psi_0}{\partial r} &= -\tilde{A}\psi_1\cancel{\cos(\vartheta)} - 2A\frac{r\cancel{\cos(\vartheta)}\beta_{Tma}}{R_0}\psi_0 \\ \frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial\psi_1}{\partial r}\right) - \frac{\psi_1}{r^2} - \frac{1}{R_0}\frac{\partial\psi_0}{\partial r} &= -\tilde{A}\psi_1 - 2A\frac{r\beta_{Tma}}{R_0}\psi_0 \\ \frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial\psi_1}{\partial r}\right) - \frac{\psi_1}{r^2} + \tilde{A}\psi_1 &= \frac{1}{R_0}\frac{\partial\psi_0}{\partial r} - 2A\frac{r\beta_{Tma}}{R_0}\psi_0 \\ \frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial\psi_1}{\partial r}\right) + \left(\tilde{A} - \frac{1}{r^2}\right)\psi_1 &= \frac{1}{R_0}\frac{\partial\psi_0}{\partial r} - 2A\frac{r\beta_{Tma}}{R_0}\psi_0\end{aligned} \tag{A.15}$$

The derivative with respect to r of Equation (A.10) is given by

$$\frac{\partial \psi_0(r)}{\partial r} = \frac{\psi_{ma}}{\pi} \int_0^\pi -\cos(\theta) \sin\left(r\sqrt{\tilde{A}}\cos(\theta)\right) \mathrm{d}\theta$$

Then substitute this result and Equation (A.10) into Equation (A.15) to get

$$\begin{aligned}
\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial \psi_1}{\partial r}\right) + \left(\tilde{A} - \frac{1}{r^2}\right)\psi_1 &= \frac{1}{R_0}\frac{\psi_{ma}}{\pi}\int_0^\pi -\cos(\theta)\sin\left(r\sqrt{\tilde{A}}\cos(\theta)\right)\mathrm{d}\theta \\
&\quad - 2A\frac{r\beta_{Tma}}{R_0}\frac{\psi_{ma}}{\pi}\int_0^\pi \cos\left(r\sqrt{\tilde{A}}\cos(\theta)\right)\mathrm{d}\theta \\
\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial \psi_1}{\partial r}\right) + \left(\tilde{A} - \frac{1}{r^2}\right)\psi_1 &= -\frac{\psi_{ma}}{R_0\pi}\int_0^\pi \left[\cos(\theta)\sin\left(r\sqrt{\tilde{A}}\cos(\theta)\right) + 2Ar\beta_{Tma}\cos\left(r\sqrt{\tilde{A}}\cos(\theta)\right)\mathrm{d}\theta\right] \\
r\frac{\mathrm{d}\psi_1}{\mathrm{d}r} + r^2\frac{\mathrm{d}^2\psi_1}{\mathrm{d}r^2} + \left(\tilde{A}r^2 - 1\right)\psi_1 &= -\frac{\psi_{ma}r^2}{R_0\pi}\int_0^\pi \left[\cos(\theta)\sin\left(r\sqrt{\tilde{A}}\cos(\theta)\right)\right. \\
&\quad \left. + 2Ar\beta_{Tma}\cos\left(r\sqrt{\tilde{A}}\cos(\theta)\right)\mathrm{d}\theta\right]
\end{aligned} \tag{A.16}$$

Now, on the left hand side is another Bessel function of order 1. It has the general (homogeneous) solution given by:

$$w(z) = J_1(z)C_1 + Y_1(z)C_2 \tag{A.17}$$

where

$$\begin{aligned}
J_1(z) &= \frac{1}{\pi}\int_0^\pi \cos\left(z\sin(\theta) - \theta\right)\mathrm{d}\theta \\
Y_1(z) &= \frac{1}{\pi}\int_0^\pi \sin\left(z\sin(\theta) - \theta\right)\mathrm{d}\theta - \frac{1}{\pi}\int_0^\infty \left[e^t - e^{-t}\right] e^{-x\sinh(t)}\,\mathrm{d}t
\end{aligned}$$

letting $z = r\sqrt{\tilde{A}}$ so that $\psi_1(r) = C_1 J_1\left(r\sqrt{\tilde{A}}\right) + C_2 Y_1\left(r\sqrt{\tilde{A}}\right)$

$$\begin{aligned}
J_1(r\sqrt{\tilde{A}}) &= \frac{1}{\pi}\int_0^\pi \cos\left(r\sqrt{\tilde{A}}\sin(\theta) - \theta\right)\mathrm{d}\theta \\
Y_1(r\sqrt{\tilde{A}}) &= \frac{1}{\pi}\int_0^\pi \sin\left(r\sqrt{\tilde{A}}\sin(\theta) - \theta\right)\mathrm{d}\theta - \frac{1}{\pi}\int_0^\infty \left[e^t - e^{-t}\right] e^{-r\sqrt{\tilde{A}}\sinh(t)}\,\mathrm{d}t
\end{aligned}$$

Do not let $C_2 = 0$ as was previously done for the ψ_0 solution. According to "Advanced Mathematical Methods for Scientists and Engineers" by Bender and Orszag, the complete homogeneous solution (meaning the solution of the first and second kind) is needed to get the inhomogeneous solution.

$$y(x) = -y_1(x)\int^x \frac{f(t)y_2(t)}{W(t)}\,\mathrm{d}t + y_2(x)\int^x \frac{f(t)y_1(t)}{W(t)}\,\mathrm{d}t$$

where the differential equation is of the form

$$Ly = f(x)$$

where

$$L = \frac{\mathrm{d}^2}{\mathrm{d}x^2} + p_1(x)\frac{\mathrm{d}}{\mathrm{d}x} + p_0(x)$$

and $W(x)$ is the Wronskian

$$W(x) = W\left[y_1(x), y_2(x)\right]$$

The $f(t)$ would be the RHS of Equation (A.16) with a change of variables for integration letting $r = t$. The Wronskian is given by

$$W\left[g_1(x), g_2(x), \ldots, g_n(x)\right] \equiv \det \begin{bmatrix} g_1(x) & g_2(x) & \cdots & g_n(x) \\ g_1'(x) & g_2'(x) & \cdots & g_n'(x) \\ \vdots & \vdots & \ddots & \vdots \\ g_1(x)^{n-1} & g_2(x)^{n-1} & \cdots & g_n^{n-1}(x) \end{bmatrix}$$

So, for this case the first derivative of the first and second kind are needed to compute the Wronskian

$$\frac{\mathrm{d}J_1(z)}{\mathrm{d}r} = J_0(z) - \frac{J_1(r)}{r}$$
$$\frac{\mathrm{d}Y_1(z)}{\mathrm{d}r} = Y_0(z) - \frac{Y_1(r)}{r}$$

So, the Wronskian is

$$\begin{aligned} W\left(r\sqrt{\tilde{A}}\right) &= W\left[C_1 J_1\left(r\sqrt{\tilde{A}}\right), C_2 Y_1\left(r\sqrt{\tilde{A}}\right)\right] \\ &= \det \begin{bmatrix} C_1 J_1\left(r\sqrt{\tilde{A}}\right) & C_2 Y_1\left(r\sqrt{\tilde{A}}\right) \\ C_1\left(J_0\left(r\sqrt{\tilde{A}}\right) - \dfrac{J_1\left(r\sqrt{\tilde{A}}\right)}{r}\right) & C_2\left(Y_0\left(r\sqrt{\tilde{A}}\right) - \dfrac{Y_1\left(r\sqrt{\tilde{A}}\right)}{r}\right) \end{bmatrix} \\ &= C_1 C_2\left(J_1\left(r\sqrt{\tilde{A}}\right) Y_0\left(r\sqrt{\tilde{A}}\right) - Y_1\left(r\sqrt{\tilde{A}}\right) J_0\left(r\sqrt{\tilde{A}}\right)\right) \end{aligned}$$

Putting this all together

$$\psi_1(r) = -C_1 J_1\left(r\sqrt{\tilde{A}}\right) \int^r \frac{f(t) C_2 Y_1\left(t\sqrt{\tilde{A}}\right)}{W(t)}\,\mathrm{d}t + C_2 Y_1\left(r\sqrt{\tilde{A}}\right) \int^r \frac{f(t) C_1 J_1\left(t\sqrt{\tilde{A}}\right)}{W(t)}\,\mathrm{d}t$$

for compactness, drop the $\left(r\sqrt{\tilde{A}}\right)$ and $\left(t\sqrt{\tilde{A}}\right)$ notation on all J and Y functions as the dependent variable can be assumed by context

$$
\begin{aligned}
&= -C_1 J_1 \int^r \frac{f(t) C_2 Y_1}{W\left[C_1 J_1, C_2 Y_1\right]} \,\mathrm{d}t + C_2 Y_1 \int^r \frac{f(t) C_1 J_1}{W\left[C_1 J_1, C_2 Y_1\right]} \,\mathrm{d}t \\
&= -\cancel{C_1} J_1 \int^r \frac{f(t) \cancel{C_2} Y_1}{\cancel{C_1 C_2}\left(J_1 Y_0 - Y_1 J_0\right)} \,\mathrm{d}t + \cancel{C_2} Y_1 \int^r \frac{f(t) \cancel{C_1} J_1}{\cancel{C_1 C_2}\left(J_1 Y_0 - Y_1 J_0\right)} \,\mathrm{d}t \\
&= -J_1 \int^r \frac{f(t) Y_1}{\left(J_1 Y_0 - Y_1 J_0\right)} \,\mathrm{d}t + Y_1 \int^r \frac{f(t) J_1}{\left(J_1 Y_0 - Y_1 J_0\right)} \,\mathrm{d}t
\end{aligned}
$$

There is one more simplification to be made thanks to Abramowitz & Stegun, the Wronskian in the denominator will reduce to $2/(\pi t)$.

$$
\psi_1(r) = \frac{\pi}{2}\left[Y_1 \int^r f(t) J_1 t \,\mathrm{d}t - J_1 \int^r f(t) Y_1 t \,\mathrm{d}t\right]
$$

There is still a major problem, however. The Y_1 represents the Bessel function of the second kind and is always divergent. This will cause the first term to diverge along with it. The second term, fortunately, does not. But what should be done about this. One solution would be so simply ignore the divergent term, but there is no justification for that. In the 0^{th} order solution I was able to do that by choosing the constant of integration to be 0. But here, the constants of integration have already canceled out. The other way of dealing with this divergent problem is to look at the limits of the integrals. Right now, I have simply used r in the upper limit as a placeholder to indicate the anti-derivative. However, If I were to set the lower limit to 0, then taking the limit $r \to 0$ both of the integrals would go to 0. Well that is all well and good, but now how do I meet the boundary condition that $\psi_1(a) = 0$? This comes from basic differential equations. A particular solution to a DE is not the only solution, of course. There is also the homogeneous solution! And here is the answer, a linear combination of the particular solution and the homogeneous solutions is also a solution! So, I simply go back to Equation (A.17) and let $C_2 = 0$. Now I can add the homogeneous solution to the particular solution to get

$$
\psi_1(r) = C_1 J_1 + \frac{\pi}{2}\left[Y_1 \int_0^r f(t) J_1 t \,\mathrm{d}t - J_1 \int_0^r f(t) Y_1 t \,\mathrm{d}t\right]
$$

And with that, substitute in for $f(t)$

$$\begin{aligned}\psi_1(r) &= C_1 J_1 \\ &\quad + \frac{\pi}{2}\left\{ Y_1 \int_0^r \frac{\psi_{ma} t^2}{R_0 \pi} \int_0^\pi \left[\cos(\theta) \sin\left(t\sqrt{\tilde{A}}\cos(\theta) \right) + 2At\beta_{Tma} \cos\left(t\sqrt{\tilde{A}}\cos(\theta) \right) \mathrm{d}\theta \right] J_1 t \, \mathrm{d}t \right. \\ &\quad \left. - J_1 \int_0^r \frac{\psi_{ma} t^2}{R_0 \pi} \int_0^\pi \left[\cos(\theta) \sin\left(t\sqrt{\tilde{A}}\cos(\theta) \right) + 2At\beta_{Tma} \cos\left(t\sqrt{\tilde{A}}\cos(\theta) \right) \mathrm{d}\theta \right] Y_1 t \, \mathrm{d}t \right\} \\ &= C_1 J_1 + \frac{\psi_{ma}}{2R_0}\left\{ Y_1 \int_0^r t^3 J_1 \int_0^\pi \left[\cos(\theta) \sin\left(t\sqrt{\tilde{A}}\cos(\theta) \right) + 2At\beta_{Tma} \cos\left(t\sqrt{\tilde{A}}\cos(\theta) \right) \mathrm{d}\theta \right] \mathrm{d}t \right. \\ &\quad \left. - J_1 \int_0^r t^3 Y_1 \int_0^\pi \left[\cos(\theta) \sin\left(t\sqrt{\tilde{A}}\cos(\theta) \right) + 2At\beta_{Tma} \cos\left(t\sqrt{\tilde{A}}\cos(\theta) \right) \mathrm{d}\theta \right] \mathrm{d}t \right\}\end{aligned}$$

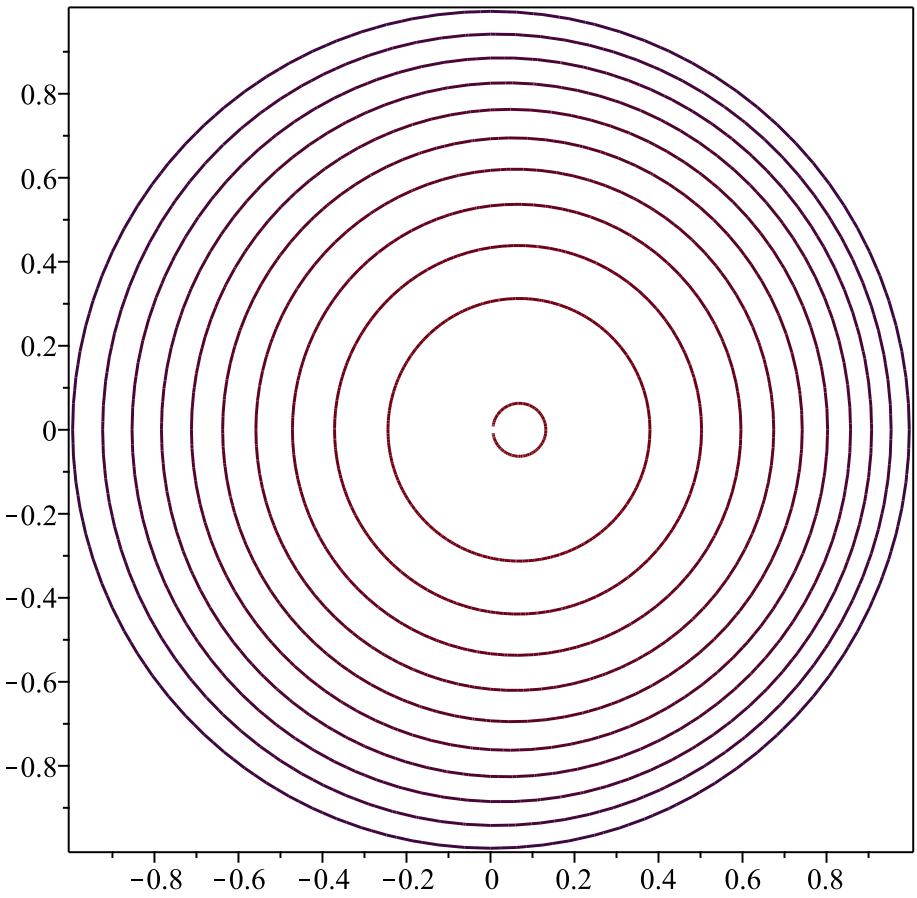
Taking the limit of this as $r \to a$ and then solving for C_1 needs to be done numerically and so a value for a needs to be assigned. For this analysis, the exact value is not important, but rather its size as compared to other values, in particular R_0. Thus, I let $a = 1$ and $R_0 = 3$ as "typical" tokamak parameters. With that, $C_1 \approxeq 0.21$ after rounding. However, I will not use this specific value in the following analysis and will instead continue to use the symbol C_1. The numeric value will be used for plotting.

A.1.3 Combining 0th & 1st Order Solutions for Plotting

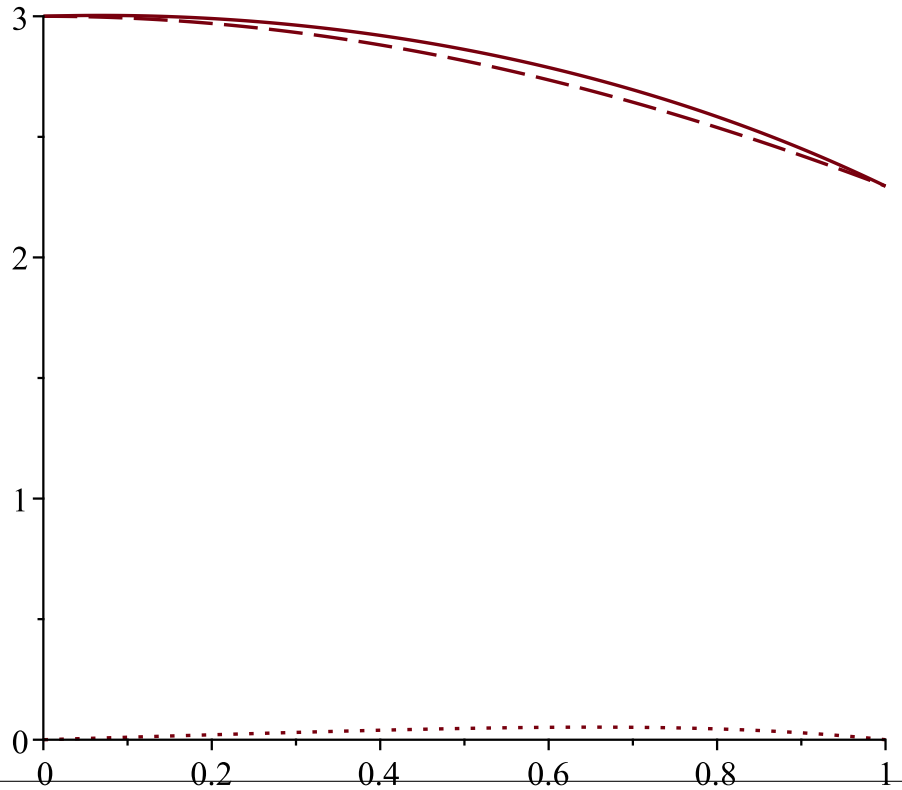
Finally, put the entire solution together and plot it as shown in Figure A.1a.

$$\begin{aligned}\psi(r,\vartheta) &= \psi_0(r) + \psi_1(r)\cos(\vartheta) \\ &= \psi_{ma} J_0 + C_1 J_1 \cos(\vartheta) \\ &\quad + \frac{\psi_{ma}}{2R_0}\left\{ Y_1 \int_0^r t^3 J_1 \int_0^\pi \left[\cos(\theta) \sin\left(t\sqrt{\tilde{A}}\cos(\theta) \right) + 2At\beta_{Tma} \cos\left(t\sqrt{\tilde{A}}\cos(\theta) \right) \mathrm{d}\theta \right] \mathrm{d}t \right. \\ &\quad \left. - J_1 \int_0^r t^3 Y_1 \int_0^\pi \left[\cos(\theta) \sin\left(t\sqrt{\tilde{A}}\cos(\theta) \right) + 2At\beta_{Tma} \cos\left(t\sqrt{\tilde{A}}\cos(\theta) \right) \mathrm{d}\theta \right] \mathrm{d}t \right\} \cos(\vartheta)\end{aligned} \tag{A.18}$$

Figure A.1b gives a visual comparison of the orders of magnitude for the $\psi_0(r)$ and $\psi_1(r)$ solutions as well as showing their sum. All the graphs below were made assuming approximate values for a tokamak such that $\frac{r}{R_0} \approx \epsilon \approx \frac{1}{3}$. Note that the shift is barely noticeable for these "typical" values. However, if $p_m a$ were increased (which would intern increase β_{Tma}), the shift would be more noticeable. For the plots below $\beta_{Tma} \sim \left(\frac{a}{R_0}\right)^2$.



(a) Countours of $\psi(r,\vartheta)$ showing the Shafranov shift due to the correction made by $\psi_1(r)\cos(\vartheta)$



A.1.4 Shafronov Shift

Using the following equation, it is possible to calculate the Shafronov shift as a function of r

$$\Delta(r) = -\psi_1(r)\left(\frac{\mathrm{d}\psi_0(r)}{\mathrm{d}r}\right)^{-1} \quad \text{where} \quad \left(\frac{\mathrm{d}\psi_0(r)}{\mathrm{d}r}\right)^{-1} = \frac{-1}{\psi_{ma}J_1}$$

$$\begin{aligned}
\Rightarrow \Delta(r) = &-C_1\cancel{J_1}\frac{1}{\psi_{ma}\cancel{J_1}} \\
&-\frac{\cancel{\psi_{ma}}}{2R_0}\left\{Y_1\int_0^r t^3 J_1 \int_0^\pi \left[\cos(\theta)\sin\left(t\sqrt{\tilde{A}}\cos(\theta)\right) + 2At\beta_{Tma}\cos\left(t\sqrt{\tilde{A}}\cos(\theta)\right)\mathrm{d}\theta\right]\mathrm{d}t\right. \\
&\qquad\qquad -\cancel{J_1}\int_0^r t^3 Y_1 \int_0^\pi \left[\cos(\theta)\sin\left(t\sqrt{\tilde{A}}\cos(\theta)\right)\right. \\
&\qquad\qquad \left.\left. + 2At\beta_{Tma}\cos\left(t\sqrt{\tilde{A}}\cos(\theta)\right)\mathrm{d}\theta\right]\mathrm{d}t\right\}\frac{1}{\cancel{\psi_{ma}}\cancel{J_1(r)}} \\
= &-\frac{C_1}{\psi_{ma}} \\
&-\frac{1}{2R_0}\left\{\frac{Y_1}{J_1}\int_0^r t^3 J_1 \int_0^\pi \left[\cos(\theta)\sin\left(t\sqrt{\tilde{A}}\cos(\theta)\right) + 2At\beta_{Tma}\cos\left(t\sqrt{\tilde{A}}\cos(\theta)\right)\mathrm{d}\theta\right]\mathrm{d}t\right. \\
&\qquad\qquad \left.-\int_0^r t^3 Y_1 \int_0^\pi \left[\cos(\theta)\sin\left(t\sqrt{\tilde{A}}\cos(\theta)\right) + 2At\beta_{Tma}\cos\left(t\sqrt{\tilde{A}}\cos(\theta)\right)\mathrm{d}\theta\right]\mathrm{d}t\right\}
\end{aligned}$$

The plot of this can be seen in Figure A.2. The maximum of value 0.07 occurs at $r = 0$ which agrees well with the magnetic axis of Figure A.1a.

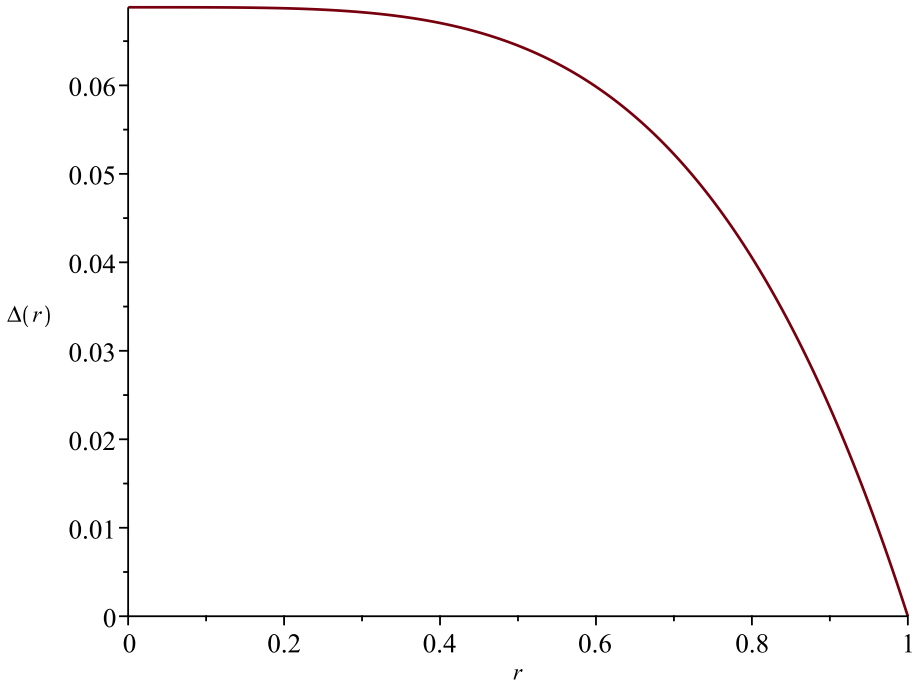


Figure A.2: The Shafranov shift expressed as $\Delta(r)$. Clearly, the maximum occurs at $r = 0$.

Appendix B

Connecting to the PCS Server

The figures below are fairly self-explanatory. All of this is done from a Windows 10 machine but should work for other versions of Windows, Apple, or Linux. The captions give further information if needed. To download and install NoMachine, follow the instructions here: `https://www.nomachine.com/download`. Now that you are connected to the PCS Virtual Desktop, you can open a terminal to begin the steps listed in Section 4.3 or Appendix E

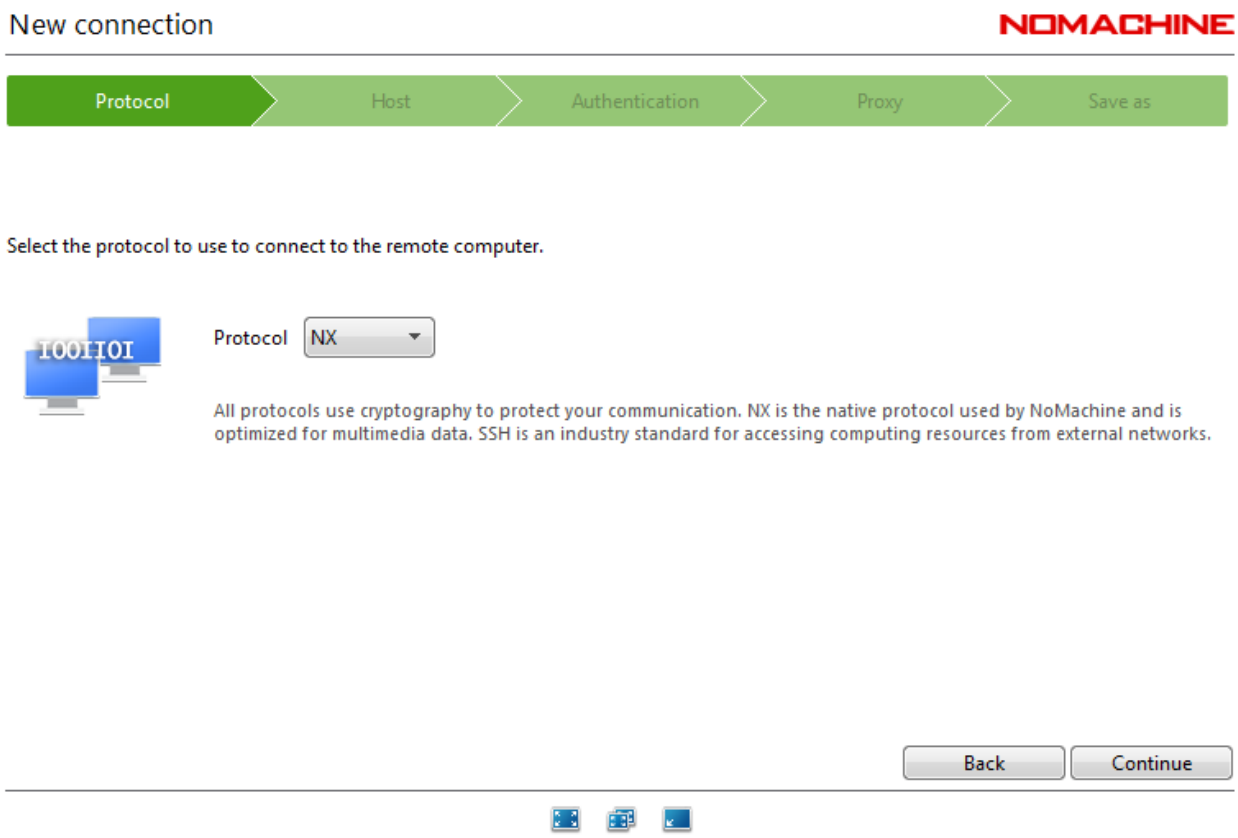


Figure B.1: Setting up NoMachine step 1: Choose the Protocol to be "NX", then Continue

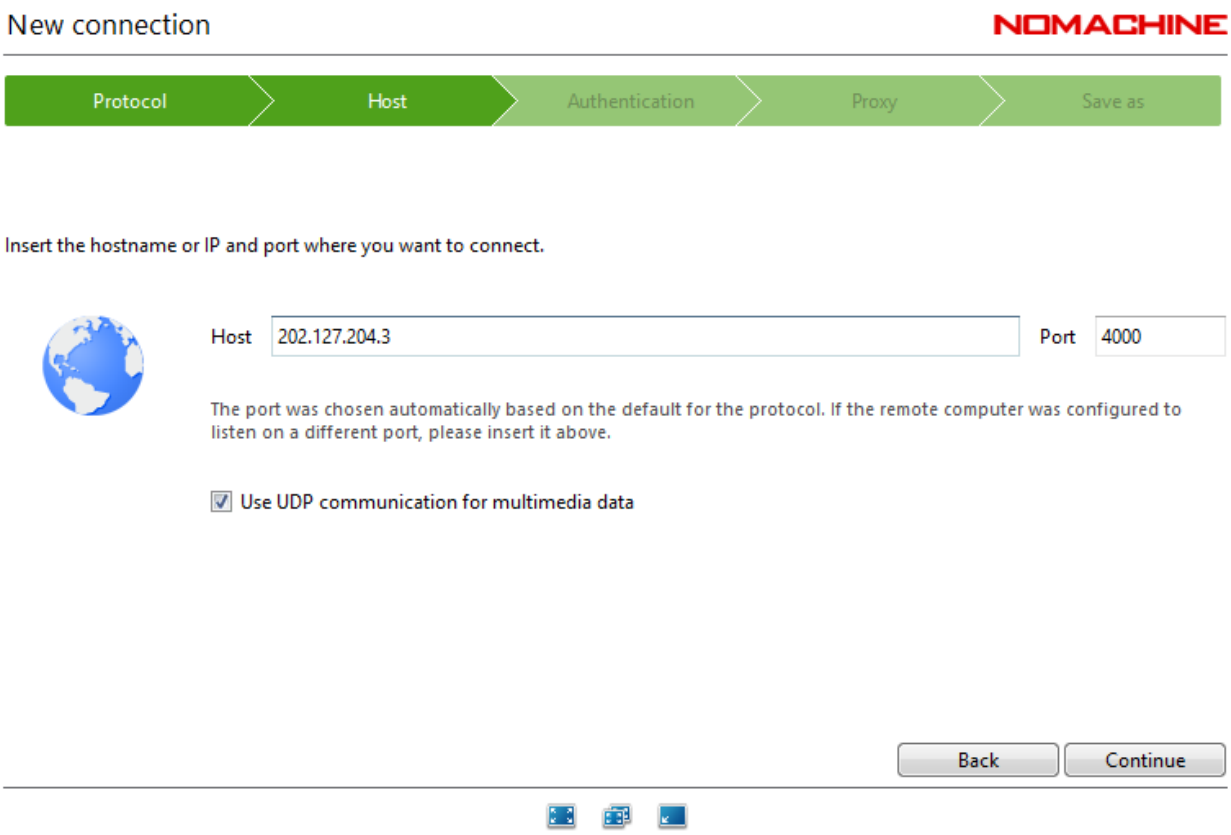


Figure B.2: Setting up NoMachine step 2: Enter the IP address 202.127.204.3, then Continue

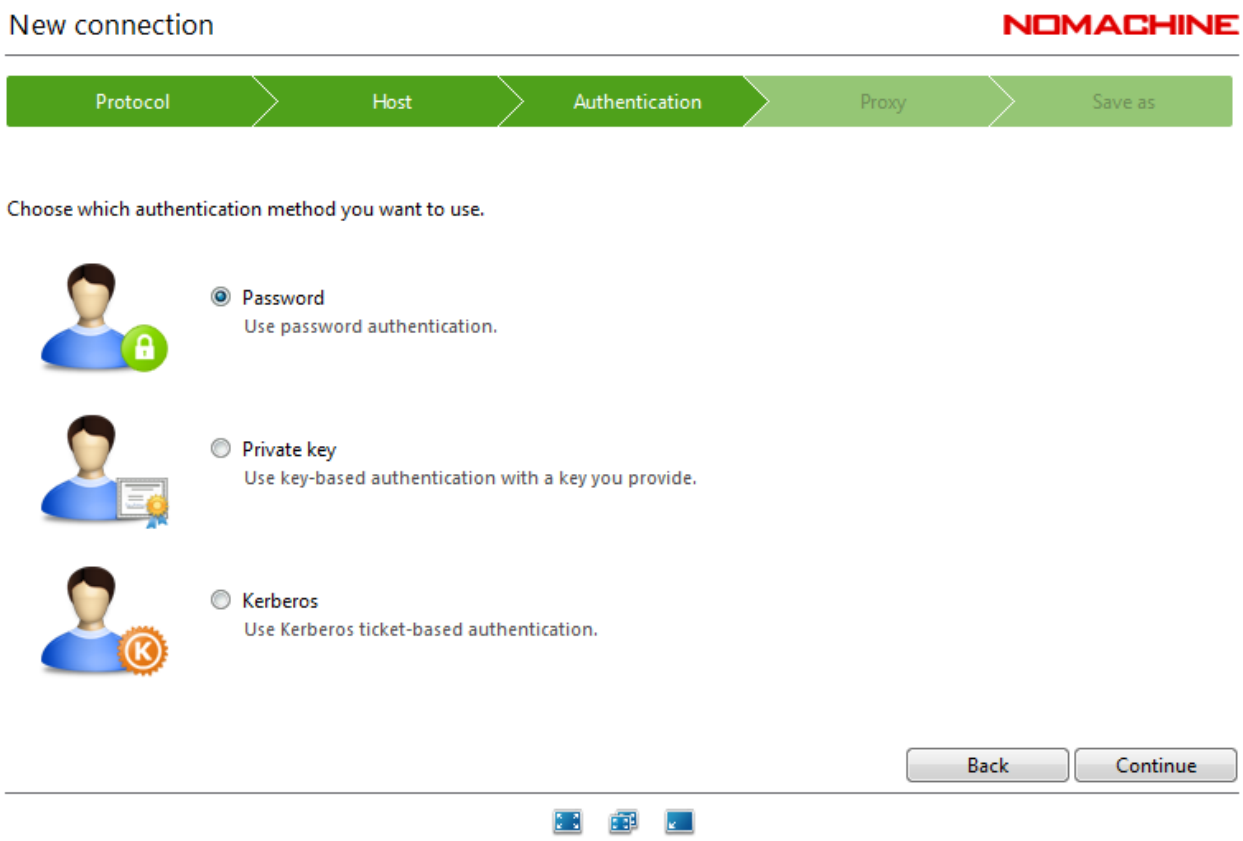


Figure B.3: Setting up NoMachine step 3: Choose the "Password" option, then Continue

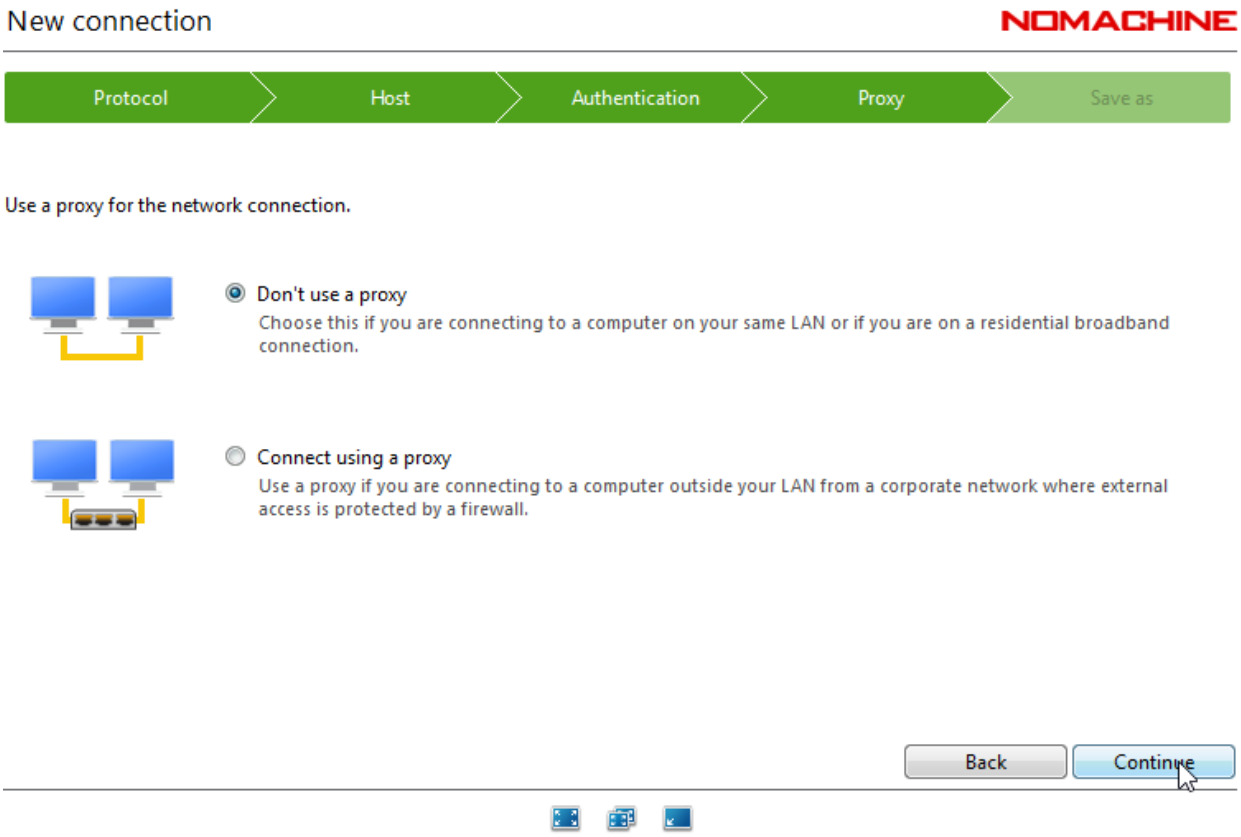


Figure B.4: Setting up NoMachine step 4: Choose the "Don't use a proxy" option, then Continue. Note, this works when using NoMachine from the EAST campus and most other places I have tried. It is possible that a proxy is needed when connecting in some locations.

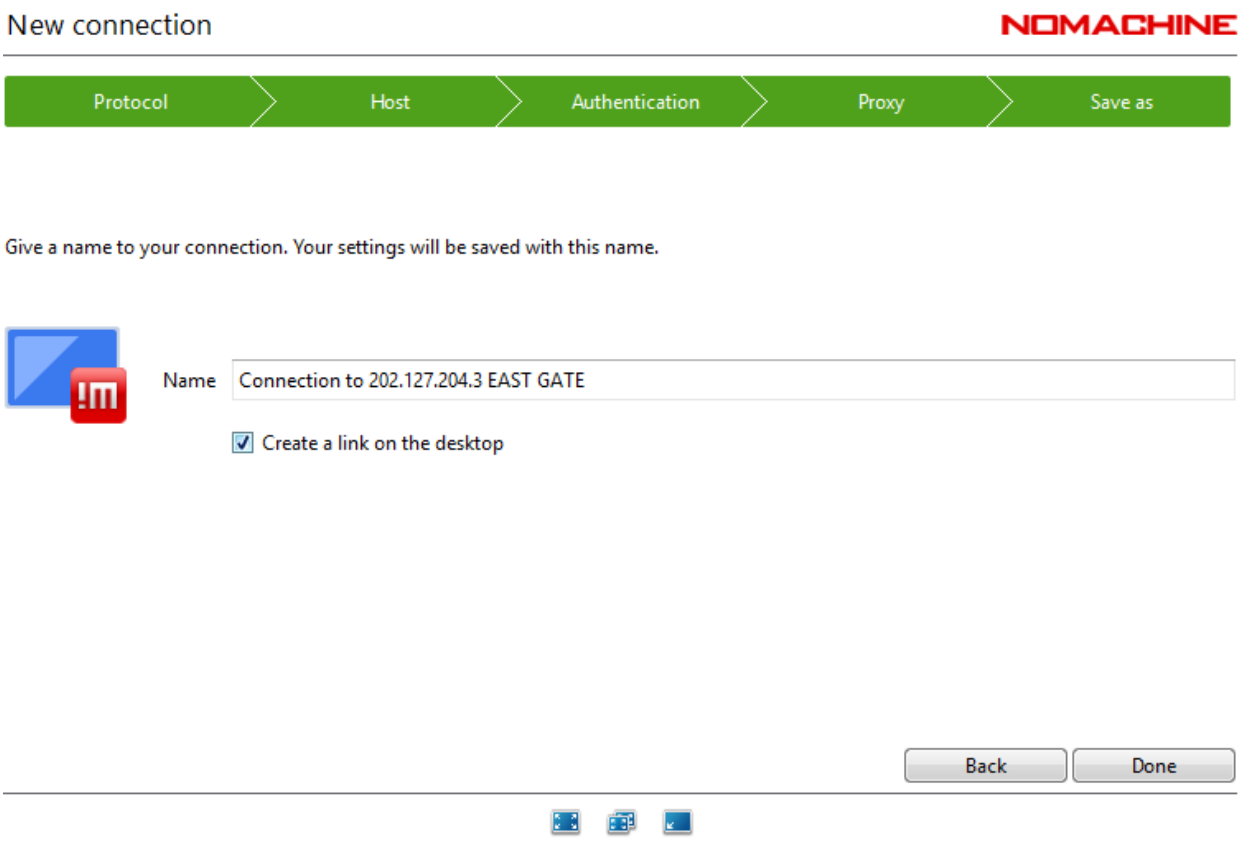


Figure B.5: Setting up NoMachine step 5: Set the connection name to whatever you like, then you are Done

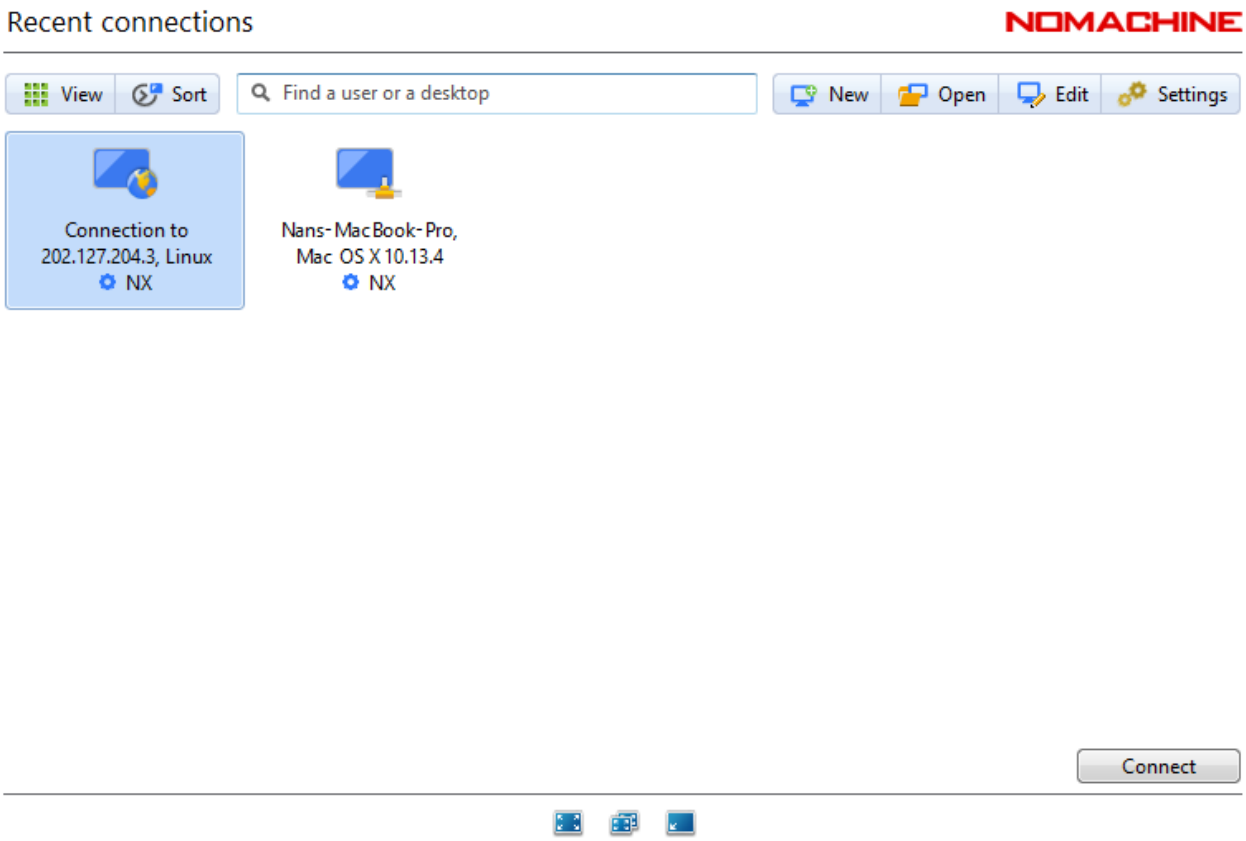


Figure B.6: Connecting to PCS server step 1: Now when you open up NoMachine you should see this screen. Choose the connection you created in the previous step, then Continue

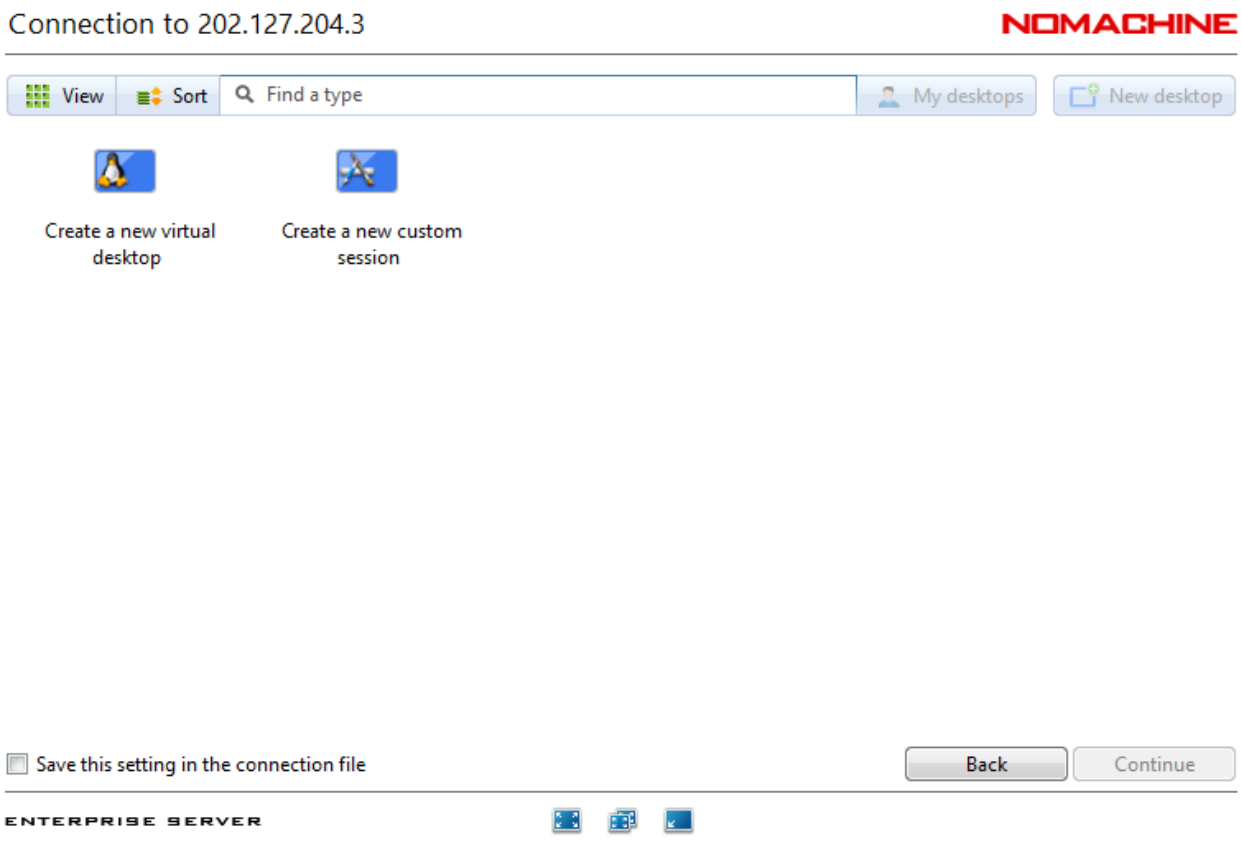


Figure B.7: Connecting to PCS server step 2: Choose the "Create a new virtual desktop", then Continue

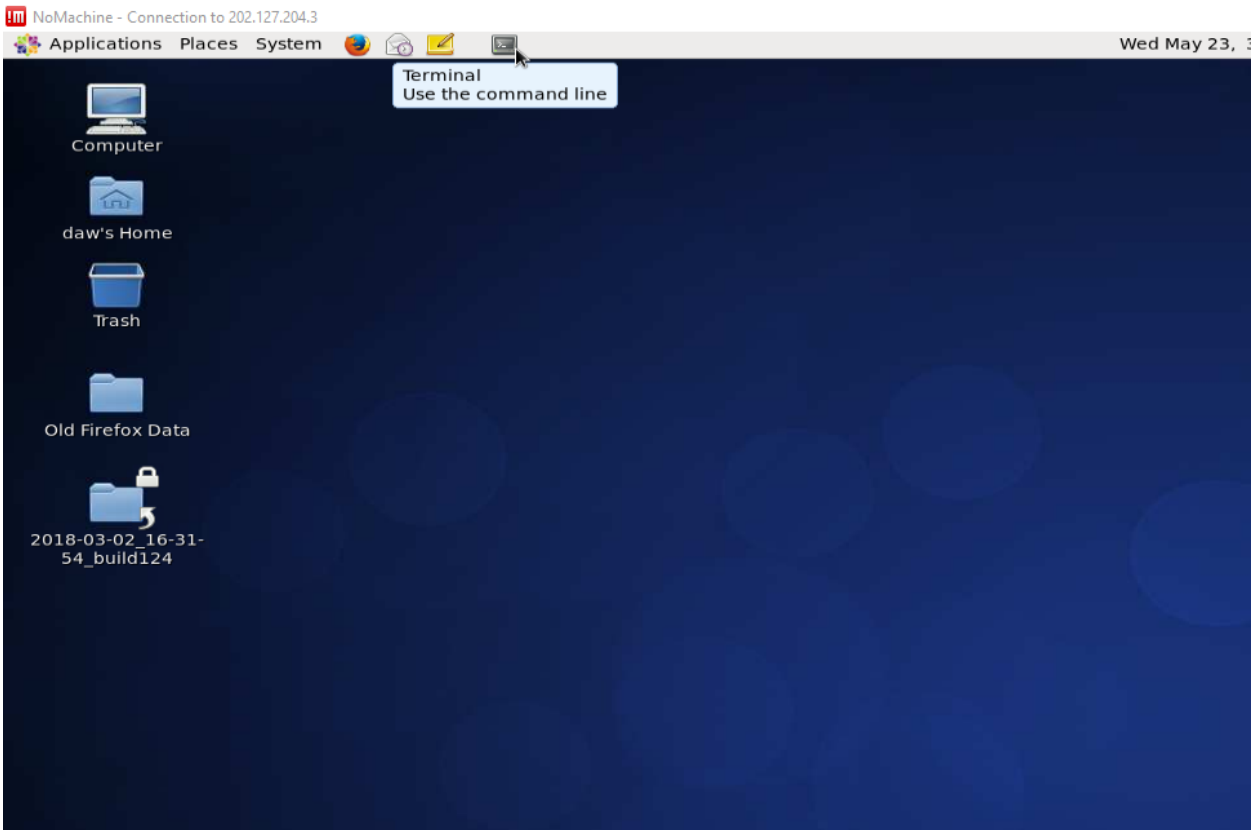


Figure B.8: After successfully connecting the PCS Virtual Desktop, open a terminal.

Appendix C

Gfile Explained

Listing C.1: Selected lines from the gfile for explanation

```
style
  EFITD      12/18/02        #  4617  ,1900ms              3  129  129
 0.140000000E+01  0.240000000E+01  0.175000000E+01  0.120000000
     E+01  0.000000000E+00
 0.187491205E+01-0.695368779E-02-0.248024932E+00-0.154744767
     E+00  0.200000000E+01
 0.184123914E+06-0.248024932E+00  0.000000000E+00  0.187491205
     E+01  0.000000000E+00
-0.695368779E-02  0.000000000E+00-0.154744767E+00  0.000000000
    E+00  0.000000000E+00
 0.351253474E+01  0.351230907E+01  0.351208589E+01  0.351186517
     E+01  0.351164692E+01
 0.350254475E+01  0.350245724E+01  0.350237149E+01  0.350228748
     E+01  0.350220520E+01
 0.118315425E+05  0.113966781E+05  0.109711236E+05  0.105547858
     E+05  0.101475715E+05
```

```
-0.114397246E+04-0.116769171E+04-0.118816356E+04-0.120548130
   E+04-0.121973819E+04
-0.109369733E+01-0.108163254E+01-0.106962319E+01-0.105766928
   E+01-0.104577081E+01
-0.125962876E+00-0.124728735E+00-0.123478622E+00-0.122213076
   E+00-0.120932629E+00
-0.154187735E+00-0.154578409E+00-0.154950512E+00-0.155303568
   E+00-0.155637175E+00
-0.155950899E+00-0.156244284E+00-0.156516851E+00-0.156768101
   E+00-0.156997524E+00
-0.157001182E+00-0.157308480E+00-0.157594355E+00-0.157858215
   E+00-0.158099456E+00
-0.147964870E+00-0.148611023E+00-0.149245952E+00-0.149869225
   E+00-0.150480389E+00
-0.151078967E+00-0.151664461E+00-0.152236347E+00-0.152794081
   E+00-0.153337094E+00
-0.153864794E+00-0.154376567E+00-0.154871779E+00-0.155349827
   E+00-0.155810299E+00
-0.156252739E+00-0.156676651E+00-0.157081500E+00-0.157466718
   E+00-0.157831703E+00
-0.160317643E+00-0.160386277E+00-0.160425716E+00-0.160435354
   E+00-0.160414617E+00
-0.160362976E+00-0.160279946E+00-0.160165089E+00-0.160018021
   E+00-0.159838409E+00
```

gfile interface description

Luo Zhengping (2007-09-27) translated by David Weldon (2017-10-30)

The Gfile file is generated by the weqdsk subroutine in the weqdeskx.for file in the efit / efitcore

directory. Output data description: (eg 65 * 65 grid point calculation result g004617.001900 example, line by line explanation). The gfile can have a maximum of 5 values per line

Line 1: EFITD 12/18/02 # ishot,itime 3 nw nh

Executed Program Date Shot Number time-slice grid size

Line 2: R direction width, Z direction width, R0 coordinate, R starting point, Z center coordinate (5 values)

Line 3: magnetic axis R coordinate, magnetic axis Z coordinate, psi value at magnetic axis, psi value at boundary, Bt (0) value

Line 4: Ip current value, psi value magnetic axis, xdum (= 0, meaningless), magnetic axis R coordinate, xdum

Line 5: Z coordinate of magnetic axis, xdum, psi value at the boundary, xdum, xdum

Line 6 - line 18: fpol — current flux function — a total of 65 (nw) nw / 5 per lines

Line 19 - Line 31: Stress — Total 65 (nw) nw / 5 per line lines

Line 32 - line 44: ffprim — ff ' value - a total of 65 (nw) nw / 5 per lines

Line 45 - line 57: pprime — pp' value - a total of 65 (nw) nw / 5 per lines

Line 58 - line 902: psi value — nw * nh the magnetic flux per point, 5 points per line, a total of nw * nh / 5 per lines

Line 903 - Line 915: qpsi — nw nw / 5 per lines

Line 916: nbbbs (number of boundary points), limitr (number of limiter points)

Line 917 - line 942: rbbbs, zbbbs (i = 1, nbbbs) - total [nw * 2/5] +1 rows

Line 943 - line 943: xlim, ylim (i = 1, limitr) - total [limitr * 2/5] +1 rows

Line 965: kvtor (intermediate variable, default = 0), rvtor (default = 1.70), nmass (intermediate variable, default = 0)

Line 966: nw, nh, ishot, itime

Line 967: rgrid (1), rgrid (nw), zgrid (1), zgrid (nh)

Line 968 - line 970: brsp (i = 1, nfcoil) - 12 poloidal field coil current values [nfcoil / 5] +1 rows

Line 971: ecurrt (i = 1, nesum nesum defaults to 1) - a total of 1, accounting for 1 line

Line 972 - line 1816: pcurrt (i = 1, nwnh) - nwnh = nw * nh total nw * nh / 5 rows per line, these are the plasma current densities.

File reading complete. All the main information is now included in the output. After this, the $OUT1 data does not need to be read.

The matlab function that reads the gfile: read_gfile_func.m

The Gfile namelist output, out1 includes: expmp2, coils, plasma, brsp, etc. are the variables containing the values of diagnostic values.

The following values are actually the calculated values from the EFIT program based on all current sources. Assign them separately before the gfile file output:

plasma =cpasma(jtime)

btor=bcentr(jtime)

do 500 i=1,nsilop

coils(i)=csilop(i,jtime)

500 continue

do 520 i=1,magpri

expmp2(i)=cmpr2(i,jtime)

520 continue

brsp is the value of the pole field coil current, which is calculated from the plasma current model coefficients and other unknown inversion parameters that have been stored during the calculation.

Appendix D

Making EAST objects

Here we need to look at the `make_east_objects.m` file found here: `/project/builds/TOKSYS/2018-03-02_16-31-54_build124/tokamaks/east/make/make_east_objects.m`. I won't include the entire script, but will instead just address a few lines of it.

I suggest that you simply save a copy of this in your EAST directory. For me, this is `/home/ASIPP/daw/EAST/make_east_objects.m`. By doing this, `Matlab` will use your modified `make_east_objects.m` file instead of the one in the path above. Whenever you reference a script, file, or variable, `Matlab` follows a preset order of places to search. Right now, your EAST directory is at the top of this search list. Then you are free to modify the script as I have done, shown in Appendix D. The comments in the code should be fairly self-explanatory. Running this script, however, may take 20 minutes or more, depending on the server.

Listing D.1: Modifications to `make_east_objects.m`

```
61 %avoid possible conflict with mpc
62 rmpath([matlabroot '/toolbox/mpc/mpc']);
63
64 % Added the 'if' statments to include gfile information DAW
       2018/04/04
65 if exist('gfile','var') && isfield(gfile,'ecase')
```

```
66      if isfield(gfile,'nh') && isfield(gfile,'nw')
67          mk_var('nr',gfile.nh);
68          mk_var('nz',gfile.nw);
69          cn = cat(2,gfile.ecase(18:21),'_',[int2str(nr) int2str(nz)
                ]);
70          mk_var('config_name',cn);
71      else
72          mk_var('nr',129);
73          mk_var('nz',129);
74          cn = cat(2,gfile.ecase(18:21),'_',[int2str(nr) int2str(nz)
                ]);
75          mk_var('config_name',cn);
76      end
77  else
78      %defaults: use rtefit grid
79      mk_var('datadir','160119');
80      mk_var('config_name','2016_3333');
81      mk_var('nr',33);
82      mk_var('nz',33);
83  end
84  mk_var('datadir','160119');
85
86  etav = 7.4e-1*ones(80,1);               %VV res uOhm-m (316SS)
87  etav(81:90)= 1.7e-2;                    % passive plates are copper
```

```
126 'dzfl', 0.001);   %height of FL for mutuals to points (like grid
        ggdata)
127
128 % Added to correct the grid using gfile information DAW 2017/09/30
129 if exist('gfile','var')
130     if isfield(gfile,'zmid') && isfield(gfile,'zdim')
131         make_tok_inputs.zgmin = gfile.zmid-gfile.zdim/2;
```

```
132         make_tok_inputs.zgmax = gfile.zmid+gfile.zdim/2;
133     end
134     if isfield(gfile,'rzero') && isfield(gfile,'xdim')
135         make_tok_inputs.rgmin = gfile.rzero-gfile.xdim/2;
136         make_tok_inputs.rgmax = gfile.rzero+gfile.xdim/2;
137     end
138     if isfield(gfile,'nh') && isfield(gfile,'nw')
139         make_tok_inputs.nr = gfile.nw;
140         make_tok_inputs.nz = gfile.nh;
141     end
142 end
143 make_tok_inputs
144 make_tok_objects(make_tok_inputs);
```

Appendix E

Matlab Preferences

To be clear, I want to take a moment to give the layout of the Matlab GUI and highlight some parts that I will refer to often in this tutorial. In Figure E.1 we see the main parts of the GUI starting at the top there is the *Toolstrip*, the *Editor* window, the *Command Window*, *Current Folder* window, and finally the *Workspace* window. Each of these windows can be moved, resized or closed within the Matlab GUI. This is my preferred layout.

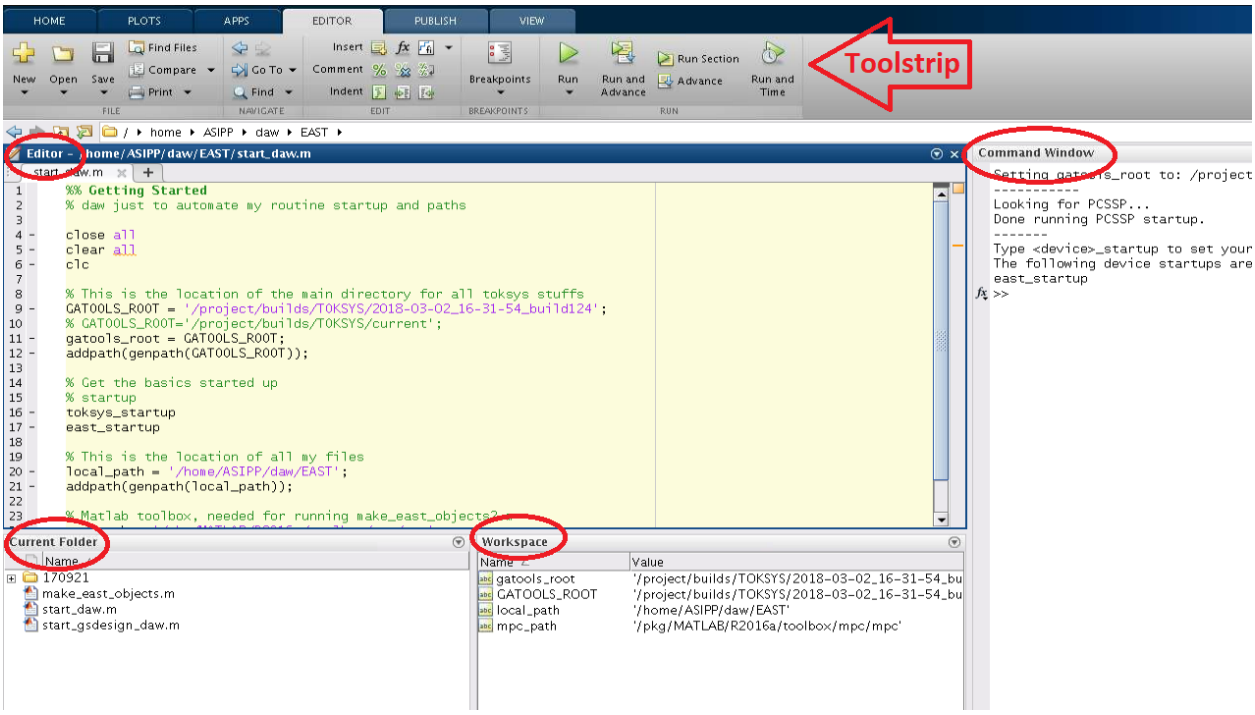


Figure E.1: Matlab layout preference

While the reader should already be familiar with the general operation of Matlab, it is also helpful to know how to change the preferences of this particular version of Matlab otherwise things such as copy & paste (ctr+c & ctr+v, respectively) and other keyboard shortcuts will not work properly. To change these go to the Matlab GUI→Home tab→Preferences→Keyboard→Shortcuts and change the desired actions to the keyboard key combinations that you prefer. If there are any conflicts then you'll need to delete them so that the keyboard shortcut works properly. See Figure E.2 - Figure E.4. Here we also need to talk about abusing the server cores. Matlab can become a very computationally intensive process when calculating a 129×129 equilibrium which will cause the server to use up to 800% of the available processing power. This is dangerous and if it lasts too long, can seriously damage the server. As you become more familiar with GSDesign and begin doing "batch" equilibria it might be necessary to use cpulimit. Below is an example of how to use it so that Matlab does not use more than 100% of the CPU. For more information, please refer to http://cpulimit.sourceforge.net/ or contact the computing administrator, Dr. Wang at wangfeng@ipp.ac.cn

Listing E.1: Changing directory, executing, the .bashrc script, logging into the server, and starting Matlab with cpulimit from the terminal.

```
[daw@node60 ~]$ cd EAST
[daw@node60 EAST]$ source .bashrc
[daw@node60 EAST]$ ssh -X cs1
daw@cs1's password:
Last login: Wed May 16 15:55:35 2018 from 202.127.205.60


************************************************************

 Welcome to EAST Computing Server 1 (CentOS6.7-64bit)

 Please contact wangfeng@ipp.ac.cn if you have any issues
```

```
13
14 ************************************************************
15
16 [daw@cs1 ~]$ cpulimit -l 100 -i matlab
17 MATLAB is selecting SOFTWARE OPENGL rendering.
```

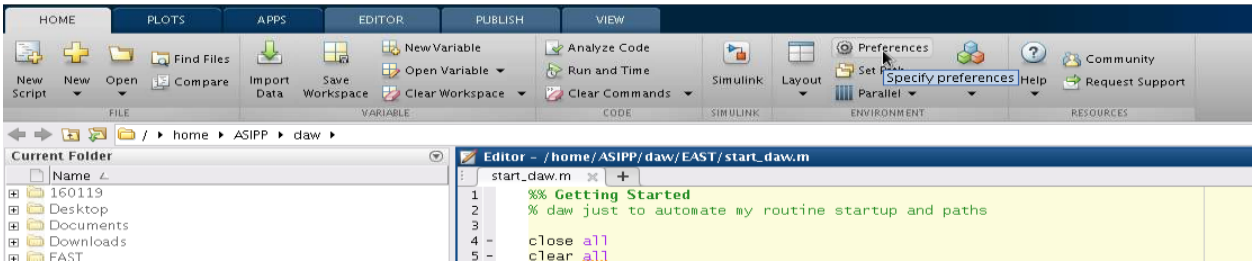


Figure E.2: Opening the `Matlab` Preferences interface

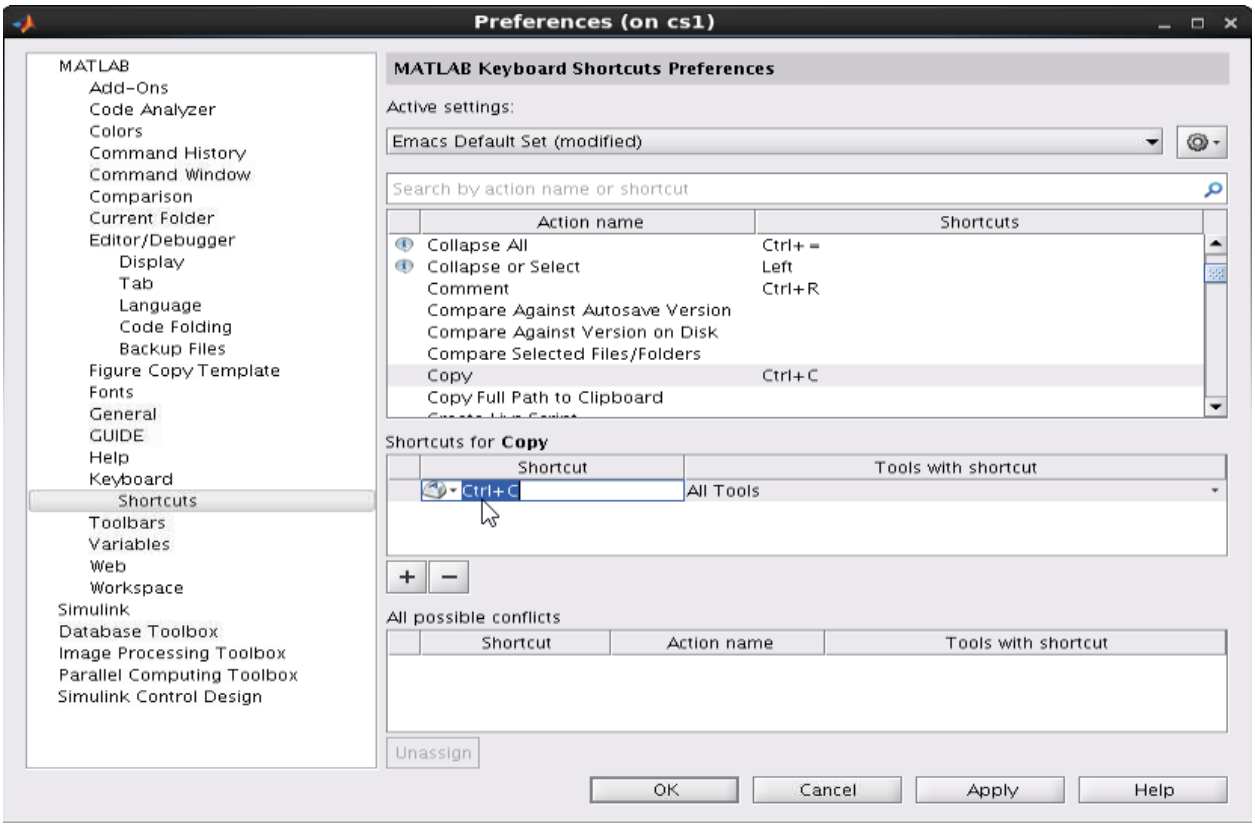


Figure E.3: Changing the keyboard shortcut for copy

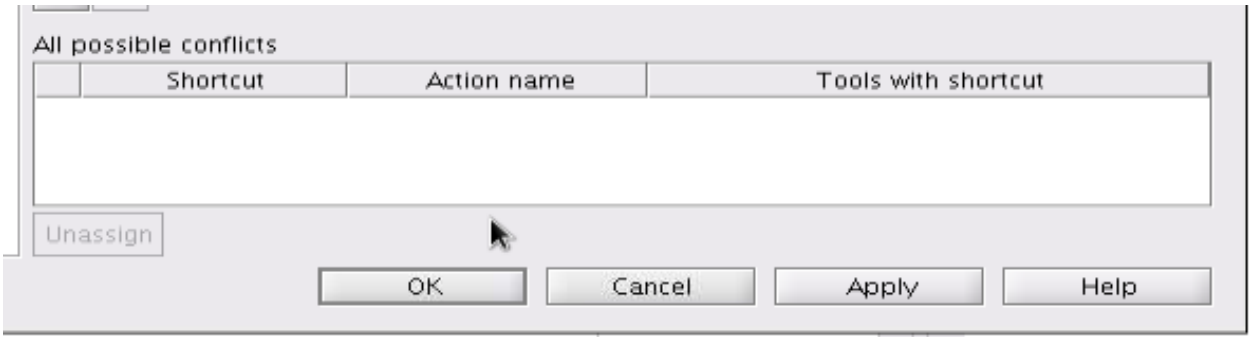


Figure E.4: Deleting any keyboard shortcut conflicts

Appendix F

GSDesign Help

This section merely provides the help documentation

Listing F.1: GSDesign help file containing definitions and some hints on strategies for getting good convergence

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

USAGE:
   eq = gsdesign(spec, init, config)

PURPOSE:
   Design a 2-D (Grad-Shafranov) equilibrium by minimizing
   cost function: norm(sum(weights.param.*(targets.param-param)))

INPUTS:

spec, specification of the equilibrium to be designed.
      This structure-variable may contain:
      targets, weights, limits, locks

      targets make up the error vector in the cost function
```

```
17      All weights = 1 by default
18      Setting a target or its weight to NaN removes it from the
            error vector
19
20      limits have two columns: lower, upper limit
21      Use -inf or +inf for no limit, ex: limits.betap = [0 inf]
22
23      locks makes a quantity equal to exactly the locked value
24      Use NaN for any quantity that should not be locked
25
26      Parameters that can be designed:
27
28      SEPARATRIX specified by: targets.rsep, targets.zsep
29      The error vector is weights.sep.*(psisep-psibry)
30      where psisep is flux at rsep, zsep
31      A (small) set of points can be locked with:
32      locks.rsep, locks.zsep, making fluxes exactly equal to psibry
33      Limits for the boundary can be set with 2-column matrices:
34      limits.rsep, limits.zsep, forcing flux to equal psibry
35      somewhere between each pair of points in columns 1 and 2.
36
37      BOUNDARY-DEFINING POINT specified by: targets.rbdef, targets.
            zbdef
38      The error vector is weights.bdef*(psibdef-psibry)
39      where psibdef is flux at targets.rbdef, targets.zbdef
40      Alternatively use locks.rbdef, locks.zbdef
41      to make psibdef at locks.rbdef, locks.zbdef exactly equal to
            psibry
42      The bdef point is prevented from jumping to other locations
            by
43      limits on flux at "wrong-way points" where bdef might
            otherwise jump.
```

```
44      The minimum normalized poloidal flux difference between such
           points
45      and the boundary is set with limits.bdef_dpsibar (default =
           0.01)
46      If limits.bdef_dpsibar is too large these other points may
47      come and go between iterations, causing convergence problems
48      The actual bdef is only guaranteed to be near rbdef, zbdef
49      If bdef is an x-point it may end up at a different poloidal
           angle
50      than rbdef, zbdef. Use targets.rx, targets.zx or locks.rx,
           locks.zx
51      if more control of bdef position is desired for a diverted
           plasma.
52
53      X-POINTS specified by: targets.rx, targets.zx
54      The error vector contains weights.x.*[dpsixdr dpsixdz]
55      for each point rx, zx, and the target is zero gradient
56      Points can be locked with: locks.rx, locks.zx
57      Limits can be specified with limits.rx and limits.zx
58      The resulting x-point for target #i will stay within
           rectangle
59      limits.rx(i,1:2), limits.zx(i,1:2). If limits.rx(i,:) has
           more
60      than 2 elements > 0 these together with limits.zx(i,:)
           specify
61      corners in a polygon and the x-point will stay within this
           polygon.
62      All x-points are shown when limits have been specified for x-
           points
63      Points with r <= 0 are ignored for targets, locks, limits.
64
65      SNOWFLAKE-POINTS specified by: targets.rsnf, targets.zsnf
```

```
66      Extra settings: targets.tsnf, targets.nsnf, targets.rhosnf
67      The flux gradient is controlled to zero at rsnf, zsnf; and
68      fluxes are controlled to equal the flux at rsnf, zsnf at nsnf
69      surrounding points a distance rhosnf away with one of those
70      being at an angle tsnf in degrees
71      Default for targets.nsnf = 6 (8,10,12,... possible if many
           coils exist)
72      Default for targets.tsnf = 40*sign(zmaxis-zsnf)
73      Default for targets.rhosnf = (dr+dz)/2
74      Default for weights.snf = 1
75      The interpolation method does not allow perfect snowflakes
           but
76      with high grid resolution perfection can be approached
77
78      SCALAR QUANTITIES
79      Targets, locks, limits can be specified for:
80        cpasma = total plasma current
81        li     = normalized inductance
82        betap  = poloidal beta
83        betan  = normalized beta
84        q0     = q at center (not yet implemented)
85        q95    = q at rhot = 0.95 (not yet implemented)
86        qmin   = q at minimum q (not yet implemented)
87        psibry = the flux at the boundary
88        psimag = the flux at the axis
89        psipla = plasma flux = current-density-weighted average
             flux
90        fluxexp = flux expansion (help calc_fluxexp) calculated by:
91           calc_fluxexp(eq, ...
92               spec.targets.rfluxexp, ...
93               spec.targets.zfluxexp, ...
94               spec.targets.dfluxexp)
```

```
95          fluxerror = maximum error in normalized poloidal flux
96            By default fluxerror is locked to 0. Including targets.
                  fluxerror = 0
97            and weights.fluxerror may improve convergence in some
                  cases.
98            Default for limits.fluxerror = [0 1e-9]
99            After cost function is minimized iterations continue
                  until
100           fluxerror < limits.fluxerror(2) and stops decreasing
101           or fluxerror < limits.fluxerror(1) which stops iterations
                   immediately
102           Set limits.fluxerror(1) > 0 to avoid iterations to
                  perfect convergence
103
104       DIAGNOSTIC SIGNALS
105       Targets, locks, limits can be specified for:
106         fl = flux loops
107         bp = magnetic probes
108         rog = rogowski loops
109       Use nan to omit elements in target and lock vectors
110         e.g. targets.bp = [4 nan 5] will set targets for bp(1) and
                bp(3)
111       Limits are size n by 2 where column 1 is min and column 2 is
              max.
112         Use -inf or +inf where there is no limit, example:
113         limits.rog = [-inf inf; 1e6 inf] to place lower limit on
                rog(2).
114
115       PROFILES ARE NOT YET IMPLEMENTED EXCEPT FOR THIS
              DOCUMENTATION
116       PROFILES (specified at points of normalized poloidal flux)
117       targets, locks, limits can be specified for:
```

```
118         pres = pressure (at nr psibar points, same as EFIT)
119         qpsi = q-profile (at nr psibar points, same as EFIT)
120       Use nan to omit elements in target and lock vectors
121       limits are size n by 2 where column 1 is min and column 2 is
              max
122         and -inf or +inf are used where there is no limit.
123         By default limits.pres = [zeros(nr,1) inf(nr,1)].
124
125       COIL CURRENTS
126       Coil currents are normally varied to minimize
127       the cost function but can also be among targets as
128         targets.ic & weights.ic
129       Specify nan for coils that have no target value.
130       To lock a coil current at a value, use locks.ic
131       Specify nan for coils that aren't locked.
132       To limit range of coil currents, use limits.ic
133       The size(limits) = [nc,2], with lower and upper limit
134       Default limits exist for DIII-D, NSTXU, KSTAR, EAST, ITER
135       Connections put extra constraints on coil currents.
136       The degrees of freedom are referred to as circuits.

137         spec.cccirc assigns a circuit (with sign) to each coil
138       Example: cccirc = [1 -1 2 3] makes 3 circuits with
139       ic(1) = -ic(2) as the first circuit current, then ic(3)
140       is the second circuit current, and ic(4) the third.
141         spec.buscode connects coils to a bus and reduces
142       number of circuits by 1 by making spec.buscode*ic = 0
143       For DIII-D an old patch panel can be loaded and used by:
144       PP = get_PP_objs(shot), spec.buscode = [0 0 PP.bus_code]
145
146       VESSEL CURRENTS
147       By default vessel currents are locked to zero, locks.iv =
```

```
        zeros(nv,1)
148     However, if spec includes a targets.iv all vessel currents
        are
149     instead unlocked by default, locks.iv = nan(nv,1)
150     The defaults can be overridden by including locks.iv in spec
151     Targets and weights for currents in vessel elements are given
         by
152       targets.iv & weights.iv
153
154     FLUX AT COILS
155     Targets for fluxes at coils can be specified by:
156       targets.psic, targets.dpsicdic, targets.ic0
157     The actual flux target for the coils are:
158     tpsic = targets.psic + targets.dpsicdic.*(ic-targets.ic0)
159     The weights for targets, locks, limits are specified with
160       weights.psic, locks.psic, limits.psic
161     By default all quantities are zero
162     One use is to design equilibrium that obeys, Vps = Rc*ic +
        Vind
163
164     FLUX AT VESSEL
165     Targets for fluxes at vessel elements can be specified by:
166       targets.psiv, targets.dpsivdiv, targets.iv0
167     The actual flux target for the coils are:
168     tpsiv = targets.psiv + targets.dpsivdiv.*(iv-targets.iv0)
169     The weights for targets, locks, limits are specified with
170       weights.psiv, locks.psiv, limits.psiv
171     By default all quantities are zero
172     One use is to design equilibrium that obeys, 0 = Rv*iv + Vind
173
174     FORCES ON COILS (frc & fzc)
175     For radial forces on coils, use:
```

```
        weights.frc, targets.frc, locks.frc, limits.frc
      For vertical forces on coils, use:
        weights.fzc, targets.fzc, locks.fzc, limits.fzc

      SWITCHES
      spec.max_iterations, overrides config.max_iterations
      spec.fig, controls what figure window gsdesign uses:
        -1 opens new figure
         0 opens new figure if last one has changed (default)
        >0 opens the figure spec.fig
      spec.showgrid, display the grid in geometry picture
      spec.showallxpoints, show all x-points
        showallxpoints=1 by default when limits specified for x-
            points
      spec.plot_settings, overrides config.plot_settings, see below

init, initial equilibrium
      The TOROIDAL FIELD is specified by init.rzero and init.bzero
      If init contains efit-specific fields these will be copied
      to the output eq to facilitate interfacing with efit codes

config, Toksys description of the tokamak (a.k.a. tok_data_struct)
        REQUIRED fields are:
          tokamak,     Name of tokamak such as 'DIII-D' or 'EAST'
          rg,          radius of grid points [m]
          zg,          height of grid points [m]
          mcc,         mutuals between coils
          mcv,         mutuals between coil and vessel elements
          mvv,         mutuals between vessel elements
          mpc,         mutuals between grid and coil elements
          mpv,         mutuals between grid and vessel elements
          mpp,         mutuals between grid points
```

```
207          limdata,     R, Z coordinates of limiter
208          imks,        1 means MKS units are used
209          iterminal,   1 means that ic gives current in 1-turn
210          fcnturn,     number of turns in the F coils
211          def_connect, used by the efit interface
212
213        OPTIONAL fields are:
214          pres0,          template pressure profile
215          pprime0,        template pprime profile, used if pres0 is
                 missing
216          fpol0,          template fpol profile
217          ffprim0,        template ffprim profile, used if fpol0 is
                 missing
218          constraints,    how profiles (pres and fpol) may vary
219                          0 = no constraint, both pres and fpol
                             vary
220                              with nkn+2 degrees of freedom
221                          1 = only 3 degrees of freedom,
222                              pres = a scaled pres0
223                              fpol = a scaled and peaked fpol0
224          psikn,          psibar for knots, default = linspace(0,1,
                nkn+1)
225          nkn,            number of knots, default = 1
226          max_iterations, default = 99, iterations stop when cost
                function
227                          has been minimized or iterations =
                             max_iterations
228                          or user clicks stop button in figure
229      plot_settings.nxptmax, maximum number of x-points in list (
             default 9)
230      plot_settings.nflux, number of flux surface contours (default
             8)
```

```
231     plot_settings.SOL.n, number of SOL contours to plot (default
            0)
232     plot_settings.SOL.d, distance between contours in outboard
            midplane (1e-3)
233     plot_settings.SymbolSize, number or struct with relative
            symbol sizes
234     Examples:
235       plot_settings.SymbolSize = 0.5; % reduce all sizes to 50%
236       plot_settings.SymbolSize.target.sep = 2; % double size of
              these
237       plot_settings.SymbolSize.wrongway = 0; % don't show wrong-
              way points
238       Symbol sizes that can be modified:
239         locks.x, locks.sep, locks.snf, locks.bdef
240         limits.x, limits.sep, limits.snf, limits.bdef
241         targets.x, targets.sep, targets.snf, targets.bdef
242         x, bdef, wrongway
243
244
245 OUTPUTS:
246
247 eq,   the designed equilibrium (eq.descriptions for more info)
248 figure with subplots:
249      Conductors: currents in coils, vessel in green, limits in red
            , blue
250      Rp',ff'/mu0R (where R=rmaxis) versus normalized poloidal flux
251      Error vector = weights.*(eq values - target values)
252        (Labels appear on error elements when zooming in on a few)
253      flux error = (psizr - psizr_pla-psizr_app)/(psimag-psibry)
254        ( < 1e-3 for good solution to the Grad-Shafranov equation)
255
256 List of scripts that demonstrate use of gsdesign:
```

```
gsdesign_demo_iter      - Design the ITER shape for ITER
gsdesign_demo_d3d_DN    - Design of EAST-like DN plasma for DIII-D
gsdesign_demo_d3d_DSNF  - Design of a double snowflake plasma for
   DIII-D
gsdesign_demo_east_LSNF - Design a lower-snowflake plasma for EAST
gsdesign_demo_east_DSNF - Design a double-snowflake plasma for EAST
gsdesign_demo_kstar_ISS - Design ITER-similar-shape (ISS) for KSTAR
gsdesign_iss            - Automatic design of ISS for 9 tokamaks
```