A Machine Learning Approach to Classifying MESSENGER FIPS Proton Spectra

Matthew K. James1, Suzanne M. Imber1,2, Jim M. Raines2, Timothy K. Yeoman1, and Emma J. Bunce1

1Department of Physics and Astronomy, University of Leicester, Leicester, UK, 2Department of Atmospheric, Oceanic and Space Sciences, University of Michigan, Ann Arbor, MI, USA

Abstract  The $\kappa$ distribution function is fitted to the entire data set of MErcury Surface, Space ENvironment, GEochemistry and Ranging’s (MESSENGER) 1-min Fast Imaging Plasma Spectrometer (FIPS Andrews et al., 2007, https://doi.org/10.1007/s11214-007-9272-5) proton spectra, and then artificial neural networks (ANNs) are used to assess the quality of this fit to the data. The $\kappa$ distribution function is fitted to each proton spectrum using the downhill-simplex method, providing an estimate for density, $n$, temperature, $T$, and the $\kappa$ parameter, which controls the shape of the distribution. The final trained neural network achieved classification accuracy of 96% and has been used to classify the 1-min proton data set collected during MESSENGER’s ~4 years in orbit of Mercury. Of the 223,282 spectra, ~160,000 were classified as having “good” fitting $\kappa$ distributions, ~133,000 of which were measurements obtained from within the magnetosphere, and ~18,000 were from the magnetosheath.

1. Introduction

The plasma mass loading of the magnetic field plays an important role in determining the time scales of processes which occur within any planetary magnetosphere. The Alfvén velocity is the characteristic speed at which magnetohydrodynamic (MHD) waves travel within a magnetized plasma and is defined by $v_A = \frac{B}{\sqrt{\mu_0 n \rho}}$, where $B$ is the magnetic field magnitude, $\mu_0$ is the permeability of free space, and $\rho$ is the plasma mass density. The plasma mass density therefore controls the propagation speed of Alfvén and magnetosonic wave modes which transmit energy, momentum, and information throughout the magnetosphere of a planet. The loading of the plasma also has an effect on magnetic reconnection rates; work by Swisdak et al. (2010) suggests that reconnection is diamagnetically suppressed when

$$\Delta \beta > \frac{2L_p}{q} \tan \frac{\theta}{2},$$

where $\Delta \beta$ is the difference in $\beta$ between the magnetosheath and magnetospheric plasmas, $\theta$ is the shear angle between the magnetic fields which thread the two fluids, and $\frac{v_A}{C_A} = 1$. For reconnection to occur during high $\Delta \beta$, the magnetic fields must be close to antiparallel (e.g., Phan et al., 2010; Swisdak et al., 2010), but when $\Delta \beta$ is low, reconnection is more favorable and is able to occur with lower shear angles (e.g., Phan et al., 2010; Poh et al., 2016). This means that mass content of a magnetized plasma is an important quantity which regulates magnetospheric time scales, response times to solar wind fluctuations, and the nature in which two plasmas may interact and is therefore a crucial factor to consider when studying magnetospheric processes.

Thus far, there have only been two missions which have sampled the Hermean plasma. Mariner 10 was able to sample the electrons in Mercury’s magnetosphere (Ogilvie et al., 1977), but there were no data collected on magnetospheric ions due to an instrument failure. The MErcury Surface, Space ENvironment, GEochemistry, and Ranging (MESSENGER Solomon et al., 2007) mission, which orbited Mercury from 18 March 2011 to 30 April 2015, hosted the FIPS instrument, one of two spectrometers which made up the Energetic Particle and Plasma Spectrometer (EPPS) instrument. FIPS was a time-of-flight instrument capable of sampling ion populations at 64 energy-per-charge ($E/q$) levels, ranging from 46 eV to 13.6 keV, every 60 s in normal mode and 10 s in burst mode (Andrews et al., 2007). While it was orbiting Mercury, FIPS obtained at least 1,250,000 individual proton spectra in the vicinity of the Hermean magnetosphere. Many of these
spectra were burst mode spectra, which were combined in groups of \( \sim 6 \) to provide 1-min average spectra and were then used to calculate plasma moments. FIPS was also capable of detecting a range of different ion species, including helium, oxygen, and sodium, the analysis of which are beyond the scope of this study.

FIPS measurements of protons and heavier ions have been used to characterize the magnetospheric (e.g., Raines et al., 2011, 2012, 2014) and magnetosheath (Gershman et al., 2013) plasmas at Mercury. These studies use plasma moments assuming a Maxwell-Boltzmann (M-B) distribution function can be used to describe the FIPS spectra, though a recent study by Sun et al. (2017) made use of the \( \kappa \) distribution to describe the plasma instead. The \( \kappa \) distribution function is similar to the Maxwellian; when \( \kappa \to \infty \), the function is equal to the Maxwellian, and when \( \kappa \) is low, the high energy tail of the distribution is enhanced and behaves like a power law. \( \kappa \) distributions which deviate significantly away from the Maxwellian distribution represent plasmas which are not in thermal equilibrium, that is, energy is being transferred to or from the plasma. Such plasma distributions are often observed in space plasmas such as the solar wind (e.g., Maksimovic et al., 1997), magnetosheath (e.g., Pollock et al., 2018), and within the plasma sheet and ring currents of planetary magnetospheres (e.g., Christon, 1987; Collier & Hamilton, 1995; Dialynas et al., 2009; Gloeckler & Hamilton, 1987; Kletzing et al., 2003; Xiao et al., 2008). Sun et al. (2017) showed that substorms in Mercury’s magnetosphere accelerate plasma sheet protons nonadiabatically, such that they deviate further from thermal equilibrium—a process which was significantly larger than similar processes measured in Earth’s magnetosphere. Another source of suprathermal plasma distributions could originate from wave-particle interactions within the magnetosphere (e.g., Summers & Ma, 2000) or in the solar wind (e.g., Miller, 1991, 1997).

The moments that are currently publicly available are all based on the Maxwellian fitting mentioned above, which does not always accurately represent the observed spectra, thus providing an inaccurate measure of density and temperature in places. Those moments do have a quality flag associated with them; the flag is based solely on there being sufficient particle counts for a fitting to be attempted, but there is no assessment of the quality of the fit. The purpose of this study is to refit the entire data set of FIPS proton spectra with the \( \kappa \) distribution function, providing improved estimates of density, \( n \), temperature, \( T \), and \( \kappa \). We take a nonlinear machine learning approach to assessing the quality of these fits to the data; artificial neural networks (ANNs) are used to determine whether or not the fit to the data is plausible, allowing us to retain the most useable FIPS spectra.

2. Data

The FIPS data products used here were downloaded from NASA’s Planetary Data System (PDS) and are arranged into three data sets: (1) Experiment Data Records (EDR), (2) Calibrated Data Records (CDR), and (3) Derived Data Records (DDR). Figure 1 shows an example of some of the products of these data sets, where the proton counts, \( C \), collected during the scan are shown in panel (a), the differential energy flux, \( \frac{dJ}{dE} \), in panel (b), and the phase space density (PSD), \( f \), in panel (c). The abscissa of each panel represents the velocity calculated for each energy-per-charge bin, where the charge is assumed to be 1 (i.e., \( v_i = \sqrt{\frac{2E_i}{m}} \), where \( i \) is the \( E/q \) bin index). The pink and gray dashed lines in each panel represent two different model fits to the data, which will be discussed in greater detail later in section 3.

The counts measured at each of the 64 \( E/q \) bins during the scan are obtained from the EDR “SCAN” data product. During FIPS’ normal operation mode, one full scan is sampled every \( \sim 60 \) s, but during burst mode, a full spectrum is provided every \( \sim 10 \) s. For the purpose of this study, the burst mode spectra are grouped into typically six (sometimes five or seven) spectra and combined such that they are directly comparable to the 60-s spectra. The counts are also used to define the fractional error using Poisson statistics,

\[
\frac{\sqrt{C_i}}{C_i}.
\]

which is used to calculate the error bars in Figure 1c, \( \delta f_i = f_i \frac{\sqrt{C_i}}{C_i} \).

The differential energy fluxes presented in Figure 1b are found in the CDR “SCAN” data product and are related to the counts by Raines (2014),

\[
\left( \frac{dJ}{dE} \right)_i = \frac{C_i}{(E/q)_i \gamma i_\eta i \delta \Omega_i}.
\]
Figure 1. Example FIPS spectral counts (a), differential energy flux (b), and PSD (c) plotted against particle velocity calculated based on the $E/q$ energy, assuming $q = 1$. The dots present in each panel represent the measured values for each $E/q$ bin, where red dots correspond to measurements of either 0 or 1 counts. The dashed gray line shows the original Maxwellian fit to the data, while the $\kappa$ distribution fit is shown as a pink dashed line. The solid black line in panel (a) shows the smoothed counts, $C_s$. 

where $\tau_i$ is the accumulation time (5 ms for burst mode and 950 ms for normal mode), the effective solid angle of the FIPS field of view $\delta\Omega = 1.15\,sr$, $\eta_i$ is the proton detection efficiency, $N_s$ is the number of scans which make up the 60-s spectrum (assuming that the counts from each energy bin, $C_i$, are summed over all scans), and $g$ is the energy-geometric factor, where $g = 8.31 \times 10^{-5}$ mm$^2$ (eV/eV) (Raines, 2014). The PSD in Figure 1c is then calculated from the differential energy flux using

$$f_i = \left(\frac{m}{v_i}\right) \left(\frac{10}{e}\right) \left(\frac{dJ}{dE}\right)_i,$$

where $m$ is the mass of the ion species in kilograms, $e$ is the elementary charge, and $v_i$ is the particle velocity corresponding to the $(E/q)$ bin $i$.

3. Refitting Spectra

The gray dashed lines in each of the panels of Figure 1 represent the fitting of an M-B function to the data provided by the “NTP” product of the DDR data set. The NTP fits have a corresponding density, $n_m$, and temperature, $T_m$, which are used to describe the overall plasma distribution.
The fits provided in the NTP data set are often a good representation of the observed proton PSDs; however, there are a number of spectra which do not fit a simple M-B distribution but would fit a $\kappa$-distribution much better. This is the case in the example shown in Figure 1, where the pink dashed lines in each panel represent the $\kappa$-distribution fit to the spectrum with a new corresponding density $n_\kappa$, temperature $T_\kappa$, and $\kappa$.

The $\kappa$-distribution function, $f_\kappa$, is given by Pierrard and Lazar (2010):

$$f_\kappa(v) = \frac{n_\kappa}{2\pi(\kappa w_\kappa^2)^{3/2}} \frac{\Gamma(\kappa + 1)}{\Gamma(\kappa - 1/2)} \left(1 + \frac{v^2}{\kappa w_\kappa^2}\right)^{-(\kappa + 1)/2},$$

$$w_\kappa = \sqrt{\frac{(2\kappa - 3)k_B T}{\kappa m}},$$

where $\Gamma(x)$ is the Gamma function. The value of $\kappa$ determines the shape of the distribution in velocity-space. High $\kappa$ distributions tend toward the M-B distribution in the limit $\kappa \to \infty$, which represents a plasma in a state of thermal equilibrium. Low $\kappa$ distributions exhibit a significant deviation from the M-B distribution with a high-energy tail and are considered to be “suprathermal.” This deviation from thermal equilibrium increases with lower values of $\kappa$ until the limit where $\kappa \to 1/2$—at which point the plasma is in a state of “anti-equilibrium” (Livadiotis & McComas, 2013) where it approaches a power law distribution. This means that the $\kappa$ distribution can be used to describe the FIPS spectra of both equilibrium (Maxwellian) and suprathermal plasmas. There are alternatives to using a single $\kappa$-distribution to describe the spectra: A suprathermal plasma could be modeled using the sum of two Maxwellians or the sum of a Maxwellian with a five parameter distribution function, $\psi$ (Livadiotis & McComas, 2009) which could be used to describe the temperature of fluids which are not in thermal equilibrium. The temperature of a $\kappa$ plasma cannot be described with the kinetic definition of temperature, using the average kinetic energy, $\langle E \rangle \equiv \frac{1}{2}m v^2$ (Maxwell, 1867). Instead, the thermodynamic definition of temperature is used (Livadiotis & McComas, 2013):

$$T \equiv \left(\frac{\partial S}{\partial U}\right)^{-1} \left[1 - \frac{1}{\kappa} \frac{S}{k_B}\right],$$

where $S$ and $U$ are the entropy and internal energy. Livadiotis and McComas (2009) showed that the definition of temperature in equation 6 with its dependence upon $\kappa$ was equivalent to the kinetic definition and could be used to describe the temperature of fluids which are not in thermal equilibrium. The temperature in 5 is related to the classical temperature, $T_0$, using

$$T = T_0 \frac{\kappa}{\kappa - 3/2},$$

where the factor of $\frac{1}{2}$ arises from the assumption of a monatomic plasma with three degrees of freedom, and $T_0$ approaches the temperature of a Maxwellian distribution when $\kappa \to \infty$. For the ease of comparison between Maxwellian and $\kappa$ distribution parameters, the temperature of the $\kappa$ distribution will be referred to as $T_\kappa$, where $T_\kappa \equiv T_0$. This means that when $\kappa$ is large, $T_\kappa \to T_m$.

Although MHD simulations have had some success modeling of Mercury’s magnetosphere (e.g., Jia et al., 2015), it is considered that kinetic effects (non-MHD) are important due to its relatively small-scale size (Baumjohann et al., 2006; Fujimoto et al., 2007) compared to the Larmor radii of the plasma ions (e.g., $r_L \approx 1,000$ km for sodium ions with $v = 400$ km s$^{-1}$ and $B = 100$ nT). This becomes even more significant as the plasma distribution deviates significantly away from thermal equilibrium as more particles exhibit high velocities. The Debye length of a suprathermal plasma, $\lambda_\kappa$, is also related to that of a Maxwellian plasma, $\lambda$, by $\lambda_\kappa \equiv \lambda(2\kappa - 3)/(2\kappa - 1)$ (Chateau & Meyer-Vernet, 1991; Pierrard & Lazar, 2010), which suggests that as $\kappa \to \infty$, $\lambda_\kappa \to \lambda$; but for lower $\kappa$ distributions, the $\lambda_\kappa < \lambda$. With a reduced Debye length at low $\kappa$, the Debye sphere contains fewer ions and is therefore less likely to behave like an ideal plasma.

Using the original density and temperatures provided by the NTP data product, ($n_m$, $T_m$), as a starting point, all of the FIPS spectra were refitted to provide new density and temperature estimates $n_\kappa$, $T_\kappa$ (here we use $T_0$ from equation 5 as it is directly comparable to $T_m$), along with a value for $\kappa$. This fitting was performed by numerically minimizing the function

$$\Delta C = \sqrt{\frac{\sum_{i=1}^{n} (C_i - C_{i,v})^2}{n}},$$

where $C_i$ is a particular fit parameter and $C_{i,v}$ the corresponding value from the NTP data product.
using the downhill simplex method (Nelder & Mead, 1965), where $C_{\kappa i}$ is the $\kappa$ distribution function converted from PSD to counts for the $i$th $E/q$ bin and $n$ is the total number of bins ($n = 64$). The value of $C_{\kappa i}$ is computed by combining equations 3 and 4 to get

$$C_{\kappa i} = r_\eta g \delta \Omega N_s \left( \frac{v_4}{20} \right) f_{\kappa i},$$

where $q = e$ and $E_i = \frac{1}{2} m v_i^2$.

Figure 2 shows a comparison between the original Maxwellian values of density (panel a) and temperature (panel b) with those obtained by fitting the $\kappa$ distribution. Each panel shows a 2D histogram of occurrence where the abscissa represents the original values of $n_m$ and $T_m$ and the ordinate is the newly fitted $n_\kappa$ and $T_\kappa$; the green dashed line in each plot represents a 1:1 ratio between the old and new fits. In Figure 2a, the densities predicted by both fits remain largely similar, with a very slight deviation toward higher $n_\kappa$ predictions when $n_m > 10$ cm$^{-3}$. The temperatures presented in Figure 2b also show that many of the new fits are close to the original values, though there is a little more spread about the 1:1 line than in panel (a). There is expected to be some difference in the temperatures predicted by the two different distribution fits. Nicolaou and Livadiotis (2016) showed that when a Maxwellian function is fitted to plasma with a $\kappa$ distribution, significant differences can arise when the plasma is not in thermal equilibrium. An examination of the $\kappa$ parameters fitted to the spectra shows that $\kappa > 10$ for approximately 73% of measurements, which means that they would be adequately described using a Maxwellian; the other 27% have low $\kappa$ and would not be well represented using a Maxwellian.

### 4. Neural Network Configuration

Ultimately, the aim for this work is to discard any FIPS spectra which cannot be used to provide a reliable estimate of the density and temperature of the plasma in the Hermean environment. This is achieved by assessing the goodness of fit between a plasma distribution model and the data, which is a straightforward task to perform manually by visual inspection when there are only a small number of spectra, but with large numbers (here 223,282 spectra), this would be a very time consuming process, with inconsistent results. Using traditional measures of goodness of fit (e.g., defining a threshold value for $\chi^2$; see section 6 for some examples) does not work very well with these data as parameters such as phase space density extend to many orders of magnitude. Instead, ANNs may be trained to classify whether or not data are reliable using a small subset of these spectra. The model then produced by the trained neural network can then be applied to the entire data set to provide a consistent classification of each spectrum. In this paper, we make use of modular neural networks, where multiple networks are trained on small sections of the spectra, the results of which are then used as inputs to a final deep neural network. This section describes in detail the architecture or the ANNs, feature selection and transformation, and neural network training.
4.1. Network Architectures

The method described here uses a total of nine “feed-forward” ANNs to determine whether the $\kappa$-distribution fit to the FIPS data is good or not. A feed-forward neural network consists of three types of layers: input, hidden, and output. These layers are comprised of “nodes,” which are somewhat analogous to neurons; each node takes the values of multiple inputs, combines them, and uses some activation function to provide an output value. The process of propagating data through a feed-forward network begins by inserting the data directly into the input layer. The output of any given layer then becomes the input to the next layer, where there is a connection between every pair of nodes in two adjacent layers (i.e., each node has an input which corresponds to the output of every node in the previous layer). The data are propagated through the network layer by layer, until the output layer is reached (this process will be discussed in further detail in section 4.4 and Appendix B). The layers which exist between the input and output layers are the hidden layers; they are only connected to the adjacent layers within the network and do not provide any visible input or output to the network.

Figure 3 shows a schematic of the overall modular network, where data are input into eight separate networks initially, the outputs of which are subsequently fed into a final network alongside other input data. The first layer of ANNs will be referred to as the “split” layer of the modular neural network, and the ANN which combines these to provide the output of the modular neural network will be referred to as the “final” layer.

In the split layer, each network assesses the quality of the model fit to the data for one eighth of the spectrum, each section comprising eight $E/q$ bins. Figure 4a depicts the architecture used for the split ANNs, where each of these networks is identical and consists of four layers, $L$, represented by shaded rectangles. The first layer is the input layer (on the left), the second and third are hidden, and the final layer is the output layer (on the right). The number of nodes in a particular layer is denoted by $s_l$, where the subscript $l$ corresponds to the layer index. The input layer, $s_0$, is defined by the number of input “features” (discussed further in section 4.2) such that $s_0 = 31$. The number of output nodes is defined by how many class labels there are for the data—each section of the spectrum must be classified as either “good” or “bad,” so $s_3 = 2$. 
Figure 4. (a) Diagram showing the architecture of the neural networks making up the split layer of the modular network (NN 0–7 of Figure 3). The blue box to the left represents the input layer to the network, where the different parameters used are listed in boxes with rounded corners, with the number of features they each provide within brackets. The yellow boxes represent the hidden layers to the network, where the number of nodes in each layer is displayed underneath. The output layer of the network, with just two nodes, is represented by a green box to the right. (b) Diagram showing the architecture of the final neural network (NN 8 in Figure 3), in the same format as that in panel (a).
Huang (2003) found that the learning capacity of a feed-forward neural network with two hidden layers can be found using

\[ s_1 = \sqrt{(K+2)N} + 2\sqrt{\frac{N}{K+2}}, \]  

(10)

and

\[ s_2 = K\sqrt{\frac{N}{K+2}}. \]  

(11)

where \( K \) is the number of output classes \( (K = s_3 = 2) \) and \( N \) is the number of samples. We choose \( N \) such that it is sufficient to model the cross-validation set, rather than the training set (both will be discussed further in section 4.3), which would implicitly add some level of regularization to the networks by reducing the total number of free parameters. Using equations 10 and 11, we find that \( s_1 = 39 \) and \( s_2 = 13 \) for the split layer networks.

The architecture of the final layer is portrayed in Figure 4b—this ANN has more inputs than that of the the split layer; there are 239 input features. The sizes of the hidden layers were determined using equations 10 and 11; layers \( s_1 \) and \( s_2 \) contain 68 and 24 nodes, respectively. The final layer has just two output nodes, as with the split layer networks.

4.2. Input Features

The input data for all nine of the networks in both split and final layers of the modular neural network are derived from the EDR and CDR data sets, or from the refitted \( \kappa \)-distributions of section 3, or a combination of both. Scalar parameters such as density, temperature, and \( \kappa \) values each provide a single “feature” of the input data, where a single feature corresponds to a single input node. Multielement parameters such as \( C \) or \( \Delta f \) (the difference in measured and modeled PSD) would provide multiple features and therefore correspond to multiple input nodes to the neural networks. The full list of parameters used in the networks are listed in Table A1, along with a brief description and the number of input nodes that they each require for the split and final ANNs.

While some of the parameters in Table A1 are either binary or restricted to values between 0 and 1, many of the parameters have a large or unlimited range of possible values. In order for the neural network to successfully utilize the input features and find an optimum of the cost function (see section 4.4) within parameter space, the input parameters should all have similar ranges in their value (e.g., Sola & Sevilla, 1997). If there is no scaling performed, then a step of a given length in parameter space, along the axis of a parameter which varies very slowly, will take a long time to find a local optimum, whereas that same step along the axis of a parameter which varies over a very short range may jump across the optimum, skipping the optimal solution entirely.

The input parameters which could have values outside of the range 0 to 1 were transformed using the Box-Cox transform (Box & Cox, 1964), such that their distributions were close to normal. The resultant normal distributions were then rescaled such that they had a mean \( \mu = 0 \) and standard deviation \( \sigma = 1 \). For a more detailed description of the Box-Cox transform and the rescaling process, see Appendix A.

4.3. Labeled Spectral Samples

For training all of the neural networks, it was necessary to manually collect and label (i.e., “good” or “bad”) a relatively small number of samples. For the split networks, a total of 1,000 spectra were split into their eight sections, each of which was manually assigned a class label of “1” (bad) or “2” (good), corresponding to the output node we wish to activate. Of these 1,000 samples, \( \sim \frac{2}{3}(667) \) were used to form the “training set” for training the neural networks; \( \sim \frac{1}{3}(167) \) formed the “cross-validation set” which is used to determine and minimize the degree of over-fitting in the networks; and \( \sim \frac{1}{6}(166) \) samples were the “test set,” which is used to provide the final measurement of accuracy for each network.

For each split section, there was an imbalance between the number of good and bad samples. A highly skewed sample set, where many more samples of one class exist than the other, could cause the neural network to be trained incorrectly: If there are 90% “good” samples, then a local optimum would exist where the network classes everything as “good,” automatically giving it an accuracy of 90%. In order to avoid this, samples with the least common class label in the training sets were replicated in order to balance the classes.
Table 1

| ANN | "Good" | "Bad" | "Good" | "Bad" |
|-----|--------|-------|--------|-------|
| 0   | 1758   | 492   | 492    | 492   |
| 1   | 287    | 380   | 380    | 380   |
| 2   | 400    | 267   | 267    | 267   |
| 3   | 407    | 260   | 407    | 407   |
| 4   | 348    | 319   | 348    | 348   |
| 5   | 256    | 411   | 411    | 411   |
| 6   | 150    | 517   | 517    | 517   |
| 7   | 80     | 587   | 587    | 587   |

(i.e., all of the samples in the smallest class were duplicated over and over until the number of samples matched the other class). Table 1 shows the number of samples of each class which form the training set for each split ANN and also the effective number of samples after the replication of spectra with the least common class label.

For the final network, NN 8 in Figure 3, a total of 3,000 labeled samples were collected, where there were exactly 1,500 of each class. The samples were split into the three sets with the same proportions as with the split layer; the training, cross-validation, and test sets contained 2,000, 500, and 500 samples, respectively, where each set had an even balance of both “good” and “bad” spectra.

For each of the aforementioned sample sets, the rescaled features of each spectrum (as described in section 4.2) are placed into a feature matrix, \( X \in \mathbb{R}^{(m \times n)} \), where \( m \) is the number of samples and \( n \) is the number of features (i.e., the number of input nodes). The class labels are also placed into a “one-hot” class matrix, \( y \in \{0, 1\}^{(m \times K)} \), where \( K \) corresponds to the total number of class labels (i.e., the number of output nodes). In this matrix, all elements are set to 0, except for the elements corresponding to the appropriate class labels for each sample, which are set to 1. The aim is to train the neural network such that, given the training set feature matrix, \( X \), it reproduces the target class matrix, \( y \), from the output layer.

4.4. Network Learning

In order for the neural network to correctly classify spectra, appropriate values for the weights and biases must be learned which correctly map the input features to the desired output class. The first step in the neural network learning process is to propagate the data from input matrix, \( X \), through the network. As previously mentioned in section 4.1, the input data are propagated through the network via a series of matrix operations and activation functions between each layer in the network—until a hypothesis matrix, \( h \), is produced at the output layer. The hypothesis matrix contains the activations of the output layer for each input sample and output node; the class label assigned to each sample is defined using the node which provides the greatest activation value. A more detailed description of neural network propagation is in Appendix B.

To assess how well the output of the neural network represents the class labels of the training data, we make use of the cross-entropy cost function,

\[
J = -\frac{1}{m} \sum_{i=1}^{m} \sum_{k=1}^{K} [y_{i,k} \log(h_{i,k}) + (1 - y_{i,k}) \log(1 - h_{i,k})] + \frac{\lambda}{2m} \sum_{l=1}^{L-1} \sum_{i=1}^{s_{l}} \sum_{j=1}^{s_{l+1}} (w_{ij}^{(l)})^2, \tag{12}
\]

where the first term provides a measure of the difference between the predefined class labels, \( y \), and the output hypothesis, \( h \), summed of \( m \) samples and \( K \) output nodes. The second term is the L2 regularization term, which uses the sum of the square of the weight matrices, \( w \), to regularize the network. L2 regularization acts to keep the weights of the network small, reducing the chance of any nodes becoming saturated, effectively smoothing the neural network response (Hagan et al., 1996, chapter 13) and thus reducing overfitting.
Learning is achieved by modifying the weight and bias matrices ($w$ and $b$, respectively) such that the cost function is minimized, corresponding to reducing the difference between the output hypothesis and the predefined class labels. Modifying the weights and biases is often done using gradient descent, where the gradients of the cost function (equation 12) with respect to each individual weight and bias are used to determine the direction in which to make a small step within “weight-space” toward a minimum in $J$. The gradients are calculated using the back-propagation algorithm (Rumelhart et al., 1985) (see Appendix C for a complete explanation). In order to train the neural networks used in this study, two gradient-descent-based algorithms with adaptive learning rates were used: For the split networks (NN 0–7 of Figure 3), we used the iRProp+ (Igel & Hüsken, 2003) algorithm for batch learning; for the final neural network (NN 8 of Figure 3), we used the RMSProp algorithm with mini-batch learning (both algorithms are discussed in more detail in Appendix C). Full batch learning uses all of the input samples to decide every single learning step, so there is one step for every pass through the input data (every “epoch”). With the final neural network, NN 8, mini-batch learning was used, where the input data are split into smaller batches and a small step is made for each batch, that is, there are multiple steps taken for each epoch (10 steps for mini-batches of 200 samples). The advantage to the mini-batch approach is that the learning algorithm is more likely to be able to jump out of a false optimum in $J$.

5. Training the FIPS Modular Network

This section describes the process used to optimize all nine ANNs which make up the FIPS modular neural network. The following six steps taken to classify the entire data set of FIPS proton spectra will be described in more detail in subsequent subsections.

1. Pretrain ANN 0–7 (split layer networks),
2. Determine optimal L2 regularization for ANN 0–7,
3. Classify split sections of entire data set of spectra,
4. Pretrain ANN 8 (final network),
5. Determine optimal L2 regularization for ANN 8, and
6. Classify all spectra.

5.1. Pretraining

The method of pretraining used in Steps 1 and 4 involves training each ANN with the training set, with no regularization. In this case, the networks soon become overfitted to the training data, with typical cross-validation accuracies in the range of 85–95%. These networks are then used to provide a very approximate or “loose” class label to the entire data set of proton spectra. The overfitting of the neural networks at this point is not an issue because the majority of spectra would still be correctly labeled, and the fine tuning of the networks can happen after the pretraining stage.

Using the ~223,000 loosely labeled spectra as a temporary training set, the networks with randomly initialized weights and biases can quickly be trained in 10 epochs, using the RMSProp algorithm with a mini-batch size of 10,000 samples. The training accuracy achieved with the loosely labeled spectra ranged from 94–98% for ANN 0–7 and was 95.3% for ANN 8. The weights and biases for each of these networks were then saved for use as a good starting point for training networks using the training sets defined in section 4.3. This process was first done for networks 0–7 and then to ANN 8 after Step 3 was completed.

5.2. Regularization and Classification

Once pretraining of a network is complete, the weights and biases found in the previous section are used to initialize neural networks in order to find the optimal regularization value, $\lambda$. Eighty-one values of $\lambda$ logarithmically spaced from $10^{-5}$–$10^{5}$ were tested for all nine neural networks. In each test of ANNs 0–7, the networks were batch trained for 200 epochs, which is enough for overfitting to occur in unregularized networks. ANN 8 was also trained for 200 epochs but using the RMSProp algorithm with mini-batches of 250 samples. When $\lambda$ is set too low, the neural networks are more susceptible to overfitting to the training data and perform badly on the cross-validation set because they have not generalized well. In the cases where $\lambda$ is set too high, the neural network is unable to converge on a solution which accurately describes the training set because regularization smooths the response of the network too much, forcing it to underfit, and therefore performs badly on both training and cross-validation sets. A good value for the L2 regularization
Table 2
Table Showing the Best \( \lambda \)s and the Classification Accuracies for Each Neural Network

| ANN | \( \lambda \)       | Accuracy (%) | Training set | Cross-validation set | Test set |
|-----|---------------------|--------------|--------------|----------------------|---------|
| 0   | \( 10^{0.1} \)      | 99.4         | 96.0         | 94.5                 |
| 1   | \( 10^{0.3} \)      | 97.5         | 96.0         | 93.8                 |
| 2   | \( 10^{-0.2} \)     | 99.3         | 95.0         | 92.7                 |
| 3   | \( 10^{0.0} \)      | 98.4         | 95.5         | 96.8                 |
| 4   | \( 10^{0.1} \)      | 97.7         | 92.0         | 92.0                 |
| 5   | \( 10^{-0.1} \)     | 97.7         | 94.4         | 94.0                 |
| 6   | \( 10^{-0.2} \)     | 96.4         | 91.6         | 91.5                 |
| 7   | \( 10^{-0.5} \)     | 99.7         | 99.3         | 96.9                 |
| 8   | \( 10^{-1.6} \)     | 96.8         | 96.6         | 96.0                 |

parameter is straightforward to find; an optimal \( \lambda \) provides a balance between underfitting and overfitting and can be characterized by a peak in cross-validation accuracy.

Table 2 shows the \( \lambda \) values found to provide the best representation of the cross-validation set for each network; the associated training, cross-validation, and test set accuracies are also provided.

The weights learned in these neural networks were then used to classify whether the fit to each section of each spectrum (Step 3), and subsequently the whole of each spectrum (Step 6), was “good” or “bad” (please see the supporting information for the final trained weight and bias matrices). Of the 223,282 spectra, 162,571 (\( \sim 73\% \)) were classified as having good fits with the \( \kappa \) distribution. Table 3 shows the numbers of good and bad spectra found at different locations around and including the magnetosphere. The regions of the space around the magnetosphere were defined using the MESSENGER MAG data to determine when the spacecraft had crossed a boundary (magnetopause or bow shock) with the method described by Winslow et al. (2013). The closed field line region in Table 3 was found by performing a magnetic field trace from the known location of MESSENGER at the time of each spectrum, using the KT17 (Korth et al., 2017) magnetic field model; any traces which had two footprints on the planet were considered to be closed.

There is one important caveat which should be restated: Because of limitations in field of view due to FIPS placement on the spacecraft, FIPS does not observe full plasma distributions in most cases and the relationship of these moments to the true plasma moments depends on some assumptions (Raines et al., 2011). The expected Mach number (M), the ratio of the bulk to the thermal speed of the plasma, provides some guidance on this matter. In the solar wind (M > 3-4), Gershman et al. (2012) showed that enough of the distribution is measured for bulk speed and temperature to be within 10% of true values, as long as some effort is made to exclude low count or non-solar wind periods (e.g., foreshock). This method should effectively make those exclusions. In sufficiently subsonic plasmas (M < 0.5), where the distribution has a much higher thermal speed than bulk speed, moments recovered from FIPS observations can be accurate provided that the distribution is reasonably isotropic. These two conditions will typically hold in the central plasma

Table 3
Table Showing the Number of Spectra Taken in Various Regions of Space Around Mercury, With a Breakdown of How Many Were Classified as Having Good and Bad Fits

| Location     | Total      | Good       | Bad        |
|--------------|------------|------------|------------|
| Magnetosphere| 192,138    | 133,762    | 58,376     |
|              | (86.05%)   | (69.62%)   | (30.38%)   |
| Magnetopause | 5,851      | 5,044      | 807        |
|              | (2.62%)    | (86.21%)   | (13.79%)   |
| Magnetosheath| 17,911     | 17,634     | 277        |
|              | (8.02%)    | (98.45%)   | (1.55%)    |
| Bow shock    | 4,701      | 4,490      | 211        |
|              | (2.11%)    | (95.51%)   | (4.49%)    |
| Solar wind   | 558        | 529        | 29         |
|              | (0.25%)    | (94.80%)   | (5.20%)    |
| Unknown      | 2,123      | 1,112      | 1,011      |
|              | (0.95%)    | (52.38%)   | (47.62%)   |
| Closed field | 131,174    | 101,053    | 30,121     |
|              | (58.75%)   | (77.04%)   | (22.96%)   |
Figure 5. Scatter plot showing the distribution of densities, \( n_\kappa \), and temperatures, \( T_\kappa \), for all “good” spectra, color-coded by the region in which the sample was taken. The magnetospheric (MS) and magnetosheath (SH) populations are the largest and are represented by blue and red points, respectively. Green points represent spectra taken during bow shock (BS) crossings, magnetopause (MP) crossing spectra are yellow, and solar wind (SW) spectra are in black.

Figure 5 shows how the “good” spectra in some of the populations in Table 3 compare in terms of their relative densities and temperatures. The magnetospheric population (MS, in blue) is by far the largest, with \( T_\kappa \) typically in the range of 1–100 MK, and \( n_\kappa \) around 0.3–30 cm\(^{-3}\). The second largest population is the magnetosheath (SH) in red; this population is lower in temperature but higher in density than that of the MS population. The spectra obtained at the magnetopause (MP) in yellow appear mostly where the MS and SH populations overlap; the MP crossings described in Winslow et al. (2013) are actually defined by the inner and outermost crossings during each pass of MESSENGER into the magnetosphere from the magnetosheath and vice versa, where MESSENGER often experienced multiple crossings through the boundary—so these spectra would most likely be of a mixture of MS or SH plasma. The bow shock (BS) and solar wind (SW) spectra are also shown in green and black, respectively, but due to the high bulk velocity, these values of \( T_\kappa \) and \( n_\kappa \) should be treated with caution.

6. Examples

Figure 6 shows nine examples of spectral fits classified using the FIPS modular neural network. In all nine panels, the PSD is plotted against the velocity, with the fitted \( \kappa \) distribution shown in pink as in Figure 1c; also shown in each of the panels are seven vertical dotted lines which separate the eight sections of the spectrum which would be classified using ANNs 0–7. The probability that each section is a “good” fit is shown at the bottom of each section, where the color is indicative of the class label: green is “good,” and red is “bad.” The overall class label output by ANN 8 is also indicated using color (with the associated probabilities shown top center)—the backgrounds of Figures 6a to 6f are shaded green to denote good overall fits; the backgrounds of Figures 6g to 6i are red, indicating that these fits were deemed to be “bad” by the final neural network. The title of each panel displays the date and time of the spectrum, with the value of \( \chi^2 \) for comparison with the ANN classification.
Figure 6. Using a similar format to that of Figure 1c, panels (a)–(f) show the phase space density of six proton spectra, each of which was classified as having a "good" fitting $\kappa$ distribution, and panels (g)–(i) show spectra classified by the neural networks as having a "bad" fit. In all panels, the vertical dotted lines represent the eight sections of the spectra, and each corresponds to a network in the split layer. The new $\kappa$-distribution fits are shown as pink dashed lines, where the gray dashed lines are the original Maxwellian fits from the NTP data set for comparison. The probabilities of each section being "good" are shown in boxes at the bottom of the panels, and the probability output from the final network is shown at the top of each panel. The green and red shading in all plots is indicative of the class label output from the network—"good" and "bad," respectively. The title of each panel shows the date and time of the spectrum, alongside the value of $\chi^2_{\kappa}$.

Panels (a)–(c) of Figure 6 each show a good fit to distributions with high $\kappa$ values which are a close match to the original Maxwellian fits in gray. The spectra in panels (a) and (b) both have four sections which were classified as being good, but both also exhibit large $\chi^2_{\kappa}$ values, whereas the spectrum in panel (c) has only two "good" sections and much lower $\chi^2_{\kappa}$ values. Panels (d)–(f) all show "good" overall fits to non-Maxwellian suprathermal plasma distributions, two of which (d and e) have very large $\chi^2_{\kappa} > 100$, where these fitted distribution functions provide very different plasma moments to those of the original NTP data set. The densities predicted in panels (d) and (e) differ by $\sim 15\%$ when compared to their Maxwellian counterparts; the most notable difference is in the temperature, which appears to be overestimated when using the M-B function to represent a non-Maxwellian spectrum. Panels (g)–(i) show three which ANN 8 considered to be "bad" fits; the spectra in panels (g) and (h) are both characterized by lower values of $\chi^2_{\kappa}$ than all six "good" spectra, which is unexpected because a good fit should be characterized by a small $\chi^2_{\kappa}$. Panel (i) shows a bad fit to a bimodal distribution, where the new $\kappa$ distribution appears to fit the higher energy of the two peaks and the original Maxwellian does not fit either peak. Bimodal distributions in FIPS proton spectra like that in Figure 6i are an unusual case, the study of which is beyond the scope of this paper, but may be of interest for further study. There is no single threshold value of $\chi^2_{\kappa}$ which could successfully divide the "good" and "bad" fits to the spectra, as demonstrated in Figure 6.
Figure 7. (a) Differential proton flux measured by FIPS during a complete pass of MESSENGER through Mercury's magnetosphere from 5:30–8:00 UT on 13 May 2012. Panel (b) shows the proton density fitted in section 3 in blue and the original density from the NTP data set in green. Panel (c) shows the fitted temperatures, in the same format as panel (b). Panel (d) shows the probability output from ANN 8 for each spectrum in red, where the class labels “good” and “bad” are indicated by the shading of the background in green and red, respectively. Finally, panel (e) shows the magnetic field as measured by MESSENGER MAG during this time, where red, green, and blue correspond to the \( x \), \( y \), and \( z \) components, and black is the total magnetic field magnitude. In all panels, the black dashed vertical line shows the time when MESSENGER passed through the magnetic equatorial plane, yellow vertical shaded areas correspond to the magnetopause crossings, and the red shaded areas represent the pass through the bow shock.
correspond to the $x$, $y$, and $z$ components of the field in the Mercury-centered solar orbital (MSO) coordinate system, and in black is the magnetic field magnitude, $\pm |\mathbf{B}|$. Present in all panels is a black, dashed vertical line which shows the time when MESSENGER crossed the magnetic equatorial plane; vertical yellow shaded areas are the times where MESSENGER crossed the magnetopause; the vertical red shaded area shows when MESSENGER crossed the bow shock. Near the start of Figure 7, at $\sim 5:30$, MESSENGER crossed the magnetopause into the southern tail lobe of the magnetosphere. At around 6:40, MESSENGER crossed the magnetic equatorial plane in Mercury's magnetotail as it traveled northward. Subsequently, MESSENGER crossed the northern cusp at around 7:15, evidenced by an enhancement in proton density and flux. Then at around 7:30, MESSENGER crossed the dayside magnetopause into the magnetosheath, where densities were much higher and temperatures much lower, and finally through the bow shock at around 7:45. The densities and temperatures predicted using our new method are not too dissimilar to the original NTP values in most cases, but in some instances in this example, densities are up to $\sim 29\%$ higher and temperatures $\sim 50\%$ lower. Generally, the new plasma moments represent a better fit to non-Maxwellian plasma distributions, and the neural networks have removed the majority of the unreliable and unusable data, as illustrated in Figure 7.

7. Conclusion

In this paper, the FIPS proton spectra have all been refitted to a $\kappa$ distribution function, which provides a better fit to non-Maxwellian plasma distributions present in the Hermean magnetosphere. This means that more accurate estimates of the plasma moments can be used to describe the Hermean plasma. Refitting of the data is coupled with a machine-learning-based method of classifying badly fitted spectra, whether the data have too few counts to provide a reasonable representation of the plasma distribution or the $\kappa$ distribution could not be used to describe the plasma (for example, if there is a bimodal velocity-space distribution).

Once the best fitting moments of the plasma are separated from the badly fitted ones, they can be used to provide accurate estimates of the typical plasma properties observed in different regions of the Hermean magnetosphere. Plasma moments are vital for a whole range of magnetospheric topics, where future work using these moments could include characterizing the variations in plasma mass densities as a function of distance along magnetic field lines, or to study the variations in the plasma properties at Mercury when extreme solar events such as interplanetary coronal mass ejections (ICMEs) occur.

The methods applied above are also not limited to the FIPS proton spectra. FIPS measurements of heavier ion populations at Mercury such as Na$^+$ would be a candidate for using this method, as would other plasma data sets, such as the measurements due to be made by the BepiColombo mission, when it reaches Mercury in 2025.

Appendix A: Input Parameter Transform

The parameters, $x$, that were not restricted within the range 0 to 1 were scaled using the Box-Cox transformation (Box & Cox, 1964),

$$
\chi^{(\lambda)} = \left\{ \begin{array}{ll}
\frac{x^{\lambda} - 1}{\lambda}, & \text{if } \lambda \neq 0, \\
\log x, & \text{if } \lambda = 0,
\end{array} \right.
$$

(A1)

where $\lambda$ is the power parameter, $x_0$ is the $x$ parameter shifted by some constant, $\delta$ such that all values are greater than zero, and $\chi^{(\lambda)}$ is the transformed parameter. The value of $\lambda$ was obtained by maximizing the log-likelihood function,

$$
f(x, \lambda) = -\frac{n}{2} \log \left[ \frac{1}{n} \sum_{i=1}^{n} \left( \frac{(x^{(\lambda)}_i - (\bar{x}^{(\lambda)})^2)}{n} \right) + (\lambda - 1) \sum_{i=1}^{n} \log(x_i) \right],
$$

(A2)

where

$$
\bar{x}^{(\lambda)} = \frac{1}{n} \sum_{i=1}^{n} x^{(\lambda)}_i,
$$

and $n$ is the number of elements in $x$. This transforms the distributions of each parameter such that they are close to normally distributed. Then each parameter is normalized to a similar scale using...
Table A1
Table of Parameters Used as to Provide Input Features

| Parameter | Features (split) | Features (final) | Description |
|-----------|------------------|------------------|-------------|
| \( C_s \) | 8                | 64               | Box-car smoothed counts, using a window length of 7 (the black line in the top panel of Figure 1). |
| \( C_V \) | 1                | 1                | Velocity corresponding to the peak in the counts, \( C \). |
| \( \sqrt{C_i} \) | 8                | 64               | Uncertainty if using Poisson statistics. |
| \( \Delta f \) | 8                | 64               | Difference between measured PSD and \( \kappa \)-distribution PSD. |
| \( \Delta f \) | 1                | 8                | Mean difference between PSD and \( \kappa \)-distribution for each split section of the spectrum. |
| \( \Delta f_{RMS} \) | 0                | 1                | RMS difference between \( f \) and \( f_{\kappa} \), weighted by the Poisson uncertainty. \( \Delta f_{RMS} = \sqrt{\frac{1}{C_i} \sum \left( \frac{C_i}{\sqrt{C_i}} \left( \log_{10} f_i - \log_{10} f_{\kappa} \right) \right)^2} \) |
| \( \chi^2 \) | 0                | 1                | \( \chi^2 = \sum \left( \frac{C_i}{\sqrt{C_i}} \left( \log_{10} f_i - \log_{10} f_{\kappa} \right) \right)^2 \). |
| \( D \) | 0                | 1                | Kolmogorov-Smirnov statistic (Smirnov, 1939). |
| \( D_{split} \) | 1                | 8                | Kolmogorov-Smirnov statistic for each split section of the spectrum. |
| \( G \) | 1                | 8                | Fraction of bins with \( C_i > 0 \). |
| \( n_\kappa \) | 1                | 1                | Density of \( \kappa \)-distribution fit. |
| \( T_\kappa \) | 1                | 1                | Temperature of \( \kappa \)-distribution fit. |
| \( \kappa \) | 1                | 1                | \( \kappa \)-distribution parameter. |
| \( y_p \) | 0                | 8                | Probability that each split section is “good,” obtained from the split layer of the modular neural networks. |
| \( y_c \) | 0                | 8                | “Good” or “bad” class label for each split section, obtained from the split layer of the modular neural networks. |

\[ x_i = \frac{x_i^{(k)} - \bar{x}_i^{(k)}}{\sigma}, \quad (A3) \]

where

\[ \sigma = \sqrt{\frac{1}{n-1} \sum_{i=1}^{n} (x_i^{(k)} - \bar{x}_i^{(k)})^2}. \]

Figure A1. Panel (a) shows the distribution of the fitted densities, \( n_\kappa \), for all of the FIPS spectra. Panel (b) shows the distribution of the densities following the Box-Cox transform.
Figure A1 shows an example of this transformation for the density parameter of all 223,282 FIPS spectra, where panel (a) shows the distribution of the original data and panel (b) shows the distribution of the transformed data. The parameters used in this transformation are shown in the upper-right corner of panel (b). The newly transformed data have been transformed into an almost Gaussian distribution, with its mean centered on 0. This step in the process of analyzing the $\kappa$ distribution fits to the FIPS data is an intermediate step, which is fully reversible (i.e., the new input data can be transformed back to their original values).

Appendix B: Forward Propagation

The data from the feature matrix are propagated through the neural networks a single layer at a time via a series of matrix operations, where the output of one layer produces the input to the next. Layer $l$ is propagated to layer $l+1$ using

$$z^{(l+1)} = a^{(l)}w^{(l)} + b^{(l)}, \quad (B1)$$

then

$$a^{(l+1)} = F(z^{(l+1)}), \quad (B2)$$

where $a^{(l)}$ is the output matrix of layer $l$ (and the input to layer $l+1$), $w^{(l)} \in \mathbb{R}^{(l)x(l+1)}$ is the weight matrix for transforming one layer to the next, $b^{(l)} \in \mathbb{R}^{1x(l+1)}$ is a bias vector, and $F$ is the activation function for any given node. As there is no layer preceding the input layer (no layer $a^{(0)}$), the input feature matrix, $X$, can be considered as the “output” of the input layer $l = 1$, that is, $X \equiv a^{(1)}$.

The activation function used for all hidden nodes in this study is the “sigmoid” function, a special case of the logistic function:

$$\sigma(z) = \frac{1}{1 + e^{-z}}, \quad (B3)$$

where an input in the range $-\infty \leq z \leq \infty$ always produces an output in the interval $0 \leq \sigma(z) \leq 1$. A related function known as the “softmax” function is used for the output layer of the neural network:

$$S(z)_j = \frac{e^j}{\sum_{k=1}^K e^k}, \quad (B4)$$

where $j$ and $k$ correspond the output nodes and $K$ is the total number of output nodes; this function always produces outputs in the range 0–1, each of which represents a probability, and the sum of which is always equal to 1. The matrix output by the final layer of the neural network, $h \in \mathbb{R}^{m \times K}$, will be referred to as the “hypothesis matrix” (Note: $h \equiv a^{(L)}$). The class labels assigned to each of the $m$ samples by the neural network is derived from $h$, where the output node with the highest probability represents the “active” output node and therefore the assigned class.

Appendix C: Back-Propagation and Training Algorithms

An efficient method for learning weights is gradient descent, which uses the gradient of the cost function, equation 12, with respect to each weight and bias (e.g., $\frac{\partial J}{\partial w^{(l)}_{ij}}$ or $\frac{\partial J}{\partial b^{(l)}_{ij}}$), and then makes a small step in the direction which would reduce $J$:

$$w^{(l)}_{ij} = w^{(l)}_{ij} - 1 \cdot \frac{\partial J}{\partial w^{(l)}_{ij}}, \quad (C1)$$

and

$$b^{(l)}_{ij} = b^{(l)}_{ij} - 1 \cdot \frac{\partial J}{\partial b^{(l)}_{ij}}, \quad (C2)$$

where $1$ is the learning rate.
The gradients of the cost function are calculated using the “back-propagation” (Rumelhart et al., 1985) algorithm, which uses the delta ($\delta$) rule to propagate errors backwards through the network. $\delta^{(l)}$ for each layer, $l$, is calculated using

$$\delta^{(l)} = \begin{cases} h - y & \text{if } l = L, \\ \delta^{(l+1)}w^T \otimes a^{(l)}(1 + a^{(l)}) & \text{otherwise}, \end{cases} \quad (C3)$$

where it is worth noting that $a^{(l)}(1 - a^{(l)})$ originates from the derivative of the activation function $\frac{\delta}{\delta a}$ or $\frac{\delta \sigma}{\delta a}$. The symbol “$\otimes$” in equation C3 is the “Hadamard” product, denoting an element-wise matrix multiplication. $\delta^{(l)}$ can then be used to calculate the gradients of $J$ with respect to the weight matrices,

$$\nabla_{w}J = \frac{1}{m} \left( (a^{(l-1)})^T \delta^{(l)} + \lambda w^{ab} \right), \quad (C4)$$

and the gradients of $J$ with respect to the biases,

$$\nabla_{b}J = \frac{1}{m} J_{1,m} \delta^{(i)}, \quad (C5)$$

where $J_{1,m}$ is a matrix of ones with the shape $1 \times m$, not to be confused with the cost function, $J$.

For training the neural networks in this study, two different but related algorithms are used, both of which are slightly more advanced than the basic gradient descent method described above in equations C1 and C2. Due to the small size of the training sets and relatively small number of input features for each of the split networks, a batch learning algorithm named iRProp+ (Igel & Hüsken, 2003) was used. For the larger final neural network and for all of the pretraining performed, a mini-batch algorithm called RMSProp was used. Both algorithms use an adaptive learning rate for each weight and bias in the neural network, which varies based on the relative signs of the weight gradients calculated during previous steps.

Using the iRProp+ algorithm, the weights and biases are updated using the following rules:

$$w_{ij} = w_{ij} - \Delta_{wi}^\gamma \text{sgn} \left( \frac{\partial J}{\partial w_{ij}} \right), \quad (C6)$$

and

$$b_{ij} = b_{ij} - \Delta_{bi}^\delta \text{sgn} \left( \frac{\partial J}{\partial b_{ij}} \right), \quad (C7)$$

where $\Delta_{wi}^\gamma$ and $\Delta_{bi}^\delta$ are the individual learning rates for the weights and biases, respectively. iRProp+ varies the step size, $\Delta_{wi}$, for a given weight or bias at step $t$ by comparing the sign of the gradient, $\frac{\partial J}{\partial w_{ij}}^{(t)}$, with the previous gradient at step $t-1$. If both gradients have the same sign, then the step size is increased multiplicatively by some value $>1$, until it reaches some maximum value $\Delta_{max}$. If the product of the gradients from step $t$ and $t+1$ is equal to 0, then there is no change to the step size. In the case where the sign has changed the step is reduced multiplicatively using a value in the range 0–1, the previous step is then reverted, and the gradient for this step is set to 0 so that during the next iteration, the step size is not updated. This algorithm is effective for batch training, where the gradients are calculated using the entire training set in one step, and only one step is taken per epoch but is not generally suitable for “online” or mini-batch training (where a subset of the input samples are used for each training step).

RMSProp is designed for use with online and mini-batch learning, where a subset of the training set is used to calculate the gradients for each step, and many steps may be taken during each training epoch. It works by keeping a moving average of the squared gradients, for example,

$$v_{ms}^{(t)} = \gamma v_{ms}^{(t-1)} + (1.0 - \gamma) \left( \frac{\partial J}{\partial w} \right)^2, \quad (C8)$$

where $\gamma$ controls the rate at which previous weight and bias gradients decay (in this case $\gamma = 0.9$). The updated rule is now

$$w_{ij} = w_{ij} - \frac{\Delta_{wi}}{\sqrt{v_{ms}} \frac{\partial J}{\partial w_{ij}}}, \quad (C9)$$

where the individual learning rates can be changed using similar rules to those for iRProp+, using $v_{ms}$ instead of $\frac{\partial J}{\partial w_{ij}}^{(t-1)}$. This algorithm stops sporadic sign changes in the weight and bias gradients from suddenly changing the direction in which the learning algorithm is moving through weight parameter space.
Data Availability Statement

The MESSENGER FIPS EDR, CDR, and DDR data sets are all available on NASA's PDS at https://pds-ppi.igpp.ucla.edu/search/?sc=Messenger&tt=Mercury&i=EPSS, as are the MAG data at https://pds-ppi.igpp.ucla.edu/search/?sc=Messenger&tt=MAG.

The plasma moments and ANN classifications made using this method are available via Figshare (https://doi.org/10.25392/leicester.data.11385897.v2) and will also be made available via NASA's PDS service. The learned weight and bias matrices and input parameter scaling factors are included both in the supporting information and in the Figshare repository.

Acknowledgments

The work by M.K.J., S.M.I., T.K.Y., and E.J.B. is supported by STFC Grant ST/H002480/1. E.J.B. is also supported by a Royal Society Wolfson Research Merit Award. J.M.R. was supported by NASA Discovery Data Analysis Program (DDAP) Grants NNX15AL01G, NNX16AJ03G, and NNX16AJ05G. This research used the ALICE High Performance Computing Facility at the University of Leicester. The MESSENGER project was also supported by the NASA Discovery Program under contracts NASW-00020 to the Carnegie Institution of Washington and NAS5-97271 to The Johns Hopkins Applied Physics Laboratory.

References

Andrews, G. B., Zurbuchen, T. H., Mauk, B. H., Malcomb, H., Fisk, L. A., Gloeckler, G., et al. (2007). The Energetic Particle and Plasma Spectrometer instrument on the MESSENGER spacecraft. Space Science Reviews, 131(1-4), 523–556. https://doi.org/10.1007/s11214-007-9272-5

Baumjohann, W., Matsuoka, A., Glassmeier, K., Russell, C., Nagai, T., Hoshino, M., et al. (2006). The magnetosphere of Mercury and its solar wind environment: Open issues and scientific questions. Advances in Space Research, 38(4), 604–609. https://doi.org/10.1016/j.asr.2005.05.117

Box, G. E., & Cox, D. R. (1964). An analysis of transformations. Journal of the Royal Statistical Society, 26(2), 211–243.

Chateau, Y. F., & Meyer-Vernet, N. (1991). Electrostatic noise in non-Maxwellian plasmas: Generic properties and kappa distributions. Journal of Geophysical Research, 96(A11), 5825–5836. https://doi.org/10.1029/90JA02565

Christen, S. (1987). A comparison of the mercury and earth magnetospheres: Electron measurements and substorm time scales. Icarus, 71(3), 448–471. https://doi.org/10.1016/0019-1335(87)90040-6

Collmer, M. R., & Hamilton, D. C. (1995). The relationship between kappa and temperature in energetic ion spectra at Jupiter. Geophysical Research Letters, 22(5), 303–306. https://doi.org/10.1029/94GL02997

Dialynas, K., Krimigis, S. M., Mitchell, D. G., Hamilton, D. C., Krupp, N., & Brandt, P. C. (2009). Energetic ion spectral characteristics in the Saturnian magnetosphere using Cassini/Mimis measurements. Journal of Geophysical Research, 114, A01212. https://doi.org/10.1029/2008JA013761

Fujimoto, M., Baumjohann, W., Kabin, K., Nakamura, R., Slavin, J. A., Terada, N., & Zelenyi, L. (2007). Hermean magnetosphere-solar wind interaction. Space Science Reviews, 132(2–4), 529–550. https://doi.org/10.1007/s11214-007-9245-8

Gershman, D. J., Slavin, J. A., Raines, J. M., Zurbuchen, T. H., Anderson, B. J., Korth, H., et al. (2013). Magnetic flux pileup and plasma depletion in Mercury’s subsolar magnetosheath. Journal of Geophysical Research: Space Physics, 118, 7181–7199. https://doi.org/10.1002/2013JA019244

Gershman, D. J., Zurbuchen, T. H., Fisk, L. A., Gilbert, J. A., Raines, J. M., Anderson, B. J., et al. (2012). Solar wind alpha particles and heavy ions in the inner heliosphere observed with MESSENGER. Journal of Geophysical Research, 117, A00M02. https://doi.org/10.1029/2012JA017829

Gloeckler, G., & Hamilton, D. C. (1987). AMPTE ion composition results. Physica Scripta, T18, 73–84. https://doi.org/10.1088/0031-8949/1987/t18/009

Hagan, M. T., Demuth, H. B., Beale, M. H., & De Jesus, O. (1996). Neural network design (Vol. 20). Boston: Pws Publishing.

Huang, G.-B. (2003). Learning capability and storage capacity of two-hidden-layer feedforward networks. IEEE Transaction on Neural Networks, 14(2), 274–281.

Igel, C., & Hüsken, M. (2003). Empirical evaluation of the improved Rprop learning algorithms. Neurocomputing, 50, 105–123. https://doi.org/10.1016/S0925-2312(01)00700-7

Jia, X., Slavin, J. A., Gombosi, T. I., Daldorff, L., Toth, G., & van der Holst, B. (2015). Global MHD simulations of Mercury’s magnetosphere with coupled planetary interior: Induction effect of the planetary conducting core on the global interaction. Journal of Geophysical Research: Space Physics, 120, 4763–4775. https://doi.org/10.1002/2015JA021143

Kletzing, C. A., Scudder, J. D., Dors, E. E., & Curto, C. (2003). Auroral source region: Plasma properties of the high-latitude plasma sheet. Journal of Geophysical Research, 108(A10), 1360. https://doi.org/10.1029/2002JA009678

Korth, H., Johnson, C. L., Philpott, L., Tsyganenko, N. A., & Anderson, B. J. (2017). A dynamic model of Mercury’s magnetospheric magnetic field. Geophysical Research Letters, 44, 10147–10154. https://doi.org/10.1002/2017GL074699

Livadiotis, G., & McComas, D. J. (2009). Beyond kappa distributions: Exploiting Tsallis statistical mechanics in space plasmas. Journal of Geophysical Research, 114, A11105. https://doi.org/10.1029/2009JA014352

Livadiotis, G., & McComas, D. J. (2013). Understanding kappa distributions: A toolbox for space science and astrophysics. Space Science Reviews, 178(1), 183–214. https://doi.org/10.1007/s11214-013-9982-9

Maksimovic, M., Pierrard, V., & Riley, P. (1997). Ultradeep electron distributions fitted with kappa functions. Geophysical Research Letters, 24(9), 1151–1154. https://doi.org/10.1029/97GL00992

Maxwell, J. C. (1867). IV. On the dynamical theory of gases. Philosophical Transactions of the Royal Society of London, 157, 49–88. https://doi.org/10.1098/rstl.1867.0004

Miller, J. A. (1991). Magnetohydrodynamic turbulence dissipation and stochastic proton acceleration in solar flares. The Astrophysical Journal, 376, 342. https://doi.org/10.1086/170284

Miller, J. A. (1997). Electron acceleration in solar flares by fast mode waves: Quasi-linear theory and pitch-angle scattering. The Astrophysical Journal, 49(2), 939–951. https://doi.org/10.1086/305004

Nelder, J. A., & Mead, R. (1965). A simplex method for function minimization. Computer Journal, 7(4), 308–313. https://doi.org/10.1093/comjnl/7.4.308

Nicolaou, G., & Livadiotis, G. (2016). Misestimation of temperature when applying Maxwellian distributions to space plasmas described by kappa distributions. Astrophysics and Space Science, 361(11), 359. https://doi.org/10.1007/s10509-016-2949-z

Ogilvie, K. W., Scudder, J. D., Vasyl’yunina, V. M., Hartle, R. E., & Siscoe, G. L. (1977). Observations at the planet Mercury by the Plasma Electron Experiment: Mariner 10. Journal of Geophysical Research, 82(13), 1807–1824. https://doi.org/10.1029/JA082i13p01807
