

---

# Understanding Pathologies of Deep Heteroskedastic Regression (Supplementary Material)

---

Eliot Wong-Toi<sup>1</sup>

Alex Boyd<sup>1</sup>

Vincent Fortuin<sup>2</sup>

Stephan Mandt<sup>1,3</sup>

<sup>1</sup>Department of Statistics, University of California, Irvine, USA

<sup>2</sup>Helmholtz AI, Munich, Germany

<sup>3</sup>Department of Computer Science, University of California, Irvine, USA

## A THEORETICAL DETAILS

### A.1 FULL FUNCTIONAL DERIVATIVES

Our FT is:

$$\mathcal{L}_{\rho, \gamma}(\hat{\mu}, \hat{\Lambda}) \approx \int_{\mathcal{X}} p(x) \rho \left[ \frac{1}{2} \hat{\Lambda}(x) \hat{r}(x)^2 - \frac{1}{2} \log \hat{\Lambda}(x) \right] + p(x) \bar{\rho} \left[ \gamma \|\nabla \hat{\mu}(x)\|_2^2 + \bar{\gamma} \|\nabla \hat{\Lambda}(x)\|_2^2 \right] dx$$

and its functional derivatives are

$$\begin{cases} \frac{\delta \mathcal{L}}{\delta \hat{\mu}} &= p(x) \rho \hat{\Lambda}(x) \hat{r}(x) - 2\bar{\rho} \gamma \Delta \hat{\mu}(x) \\ \frac{\delta \mathcal{L}}{\delta \hat{\Lambda}} &= \frac{p(x) \rho}{2} \left[ \hat{r}(x)^2 - \frac{1}{\hat{\Lambda}(x)} \right] - 2\bar{\rho} \bar{\gamma} \Delta \hat{\Lambda}(x), \end{cases} \quad (11)$$

where  $\hat{r}(x) = y(x) - \hat{\mu}(x)$ . After setting equal to zero we arrive at

$$\begin{cases} \hat{\Lambda}^*(x) (\hat{\mu}^*(x) - y(x)) = 2 \frac{\bar{\rho}}{\rho} \gamma \frac{\Delta \hat{\mu}^*(x)}{p(x)} \\ (y(x) - \hat{\mu}^*(x))^2 = \frac{1}{\hat{\Lambda}^*(x)} + 4 \frac{\bar{\rho}}{\rho} \bar{\gamma} \frac{\Delta \hat{\Lambda}^*(x)}{p(x)}. \end{cases} \quad (12)$$

### A.2 PROOFS

**Proposition 1.** *Assuming there exists twice differentiable functions  $\mu : \mathbb{R}^d \rightarrow \mathbb{R}$ ,  $\Lambda : \mathbb{R}^d \rightarrow \mathbb{R}_{>0}$ , the following properties hold*

- i In the absence of regularization ( $\rho = 1$ ), there are no solutions to the FT.*
- ii In the absence of data ( $\rho = 0$ ), there is no unique solution to the FT.*
- iii In order for there to exist a solution to the FT there must be regularization on the mean function.*

*Proof.* Without loss of generality, we consider a uniform  $p(x)$  and drop it from the equations.

(i) When  $\rho = 1$  the necessary conditions for an optima are

$$\begin{cases} \hat{\Lambda}^*(x)(\hat{\mu}^*(x) - y(x)) = 0 \\ (\hat{\mu}^*(x) - y(x))^2 = \frac{1}{\hat{\Lambda}^*(x)} \end{cases} \quad (13)$$

$$\implies \begin{cases} \hat{\Lambda}^*(x)(\hat{\mu}^*(x) - y(x)) = 0 \\ \hat{\Lambda}^*(x)(\hat{\mu}^*(x) - y(x))^2 = 1 \end{cases} \quad (14)$$

$$\implies \begin{cases} \hat{\Lambda}^*(x)(\hat{\mu}^*(x) - y(x)) = 0 \\ 0 \times (\hat{\mu}^*(x) - y(x)) = 1 \end{cases} \quad (15)$$

$$\implies 0 = 1 \quad (16)$$

which is a contradiction and there cannot exist  $\mu, \Lambda$  that are solutions.

(ii) When  $\rho = 0$  the integral we seek to maximize is:

$$\mathcal{L}_{\rho, \gamma}(\hat{\mu}, \hat{\Lambda}) = \int_{\mathcal{X}} \rho \int_{\mathcal{Y}} p(y|x) \log \hat{p}(y|x) dy + \bar{\rho} \left[ \gamma \|\nabla \hat{\mu}(x)\|_2^2 + \bar{\gamma} \|\nabla \hat{\Lambda}(x)\|_2^2 \right] dx \quad (17)$$

$$= \int_{\mathcal{X}} \left[ \gamma \|\nabla \hat{\mu}(x)\|_2^2 + \bar{\gamma} \|\nabla \hat{\Lambda}(x)\|_2^2 \right] dx \quad (18)$$

where we  $\hat{p}(y|x) = \mathcal{N}(y|\hat{\mu}(x), \hat{\Lambda}(x))$ . Each term in this integral is non-negative, so the minimum value it could be is zero. Any pair of constant functions  $\mu, \Lambda$  will minimize this integral, of which there are infinitely many.

(iii) In the  $(\alpha, \beta)$ -regularization, it is equivalent to say  $\alpha > 0$  is a necessary condition for there to exist a solution to the FT. Recall that we seek to minimize

$$\mathcal{L}_{\alpha, \beta}(\hat{\mu}, \hat{\Lambda}) = \int_{\mathcal{X}} p(x)(-\log \hat{p}(y|x) + \alpha \|\nabla \hat{\mu}(x)\|_2^2 + \beta \|\nabla \hat{\Lambda}(x)\|_2^2) dx$$

where we  $\hat{p}(y|x) = \mathcal{N}(y|\hat{\mu}(x), \hat{\Lambda}(x))$ . Suppose  $\alpha = 0$ . Then the functional simplifies to

$$\min_{\hat{\mu}, \hat{\Lambda}} \mathcal{L}_{\alpha, \beta}(\hat{\mu}, \hat{\Lambda}) = \min_{\hat{\mu}, \hat{\Lambda}} \int_{\mathcal{X}} p(x)(-\log \hat{p}(y|x) + \beta \|\nabla \hat{\Lambda}(x)\|_2^2) dx \quad (19)$$

$$\leq \min_{\hat{\mu}, \hat{\Lambda}} \int_{\mathcal{X}} p(x) \frac{1}{2} (\hat{\Lambda}(x) \hat{r}(x)^2 - \log \hat{\Lambda}(x)) + p(x) \beta \|\nabla \hat{\Lambda}(x)\|_2^2 dx \quad (20)$$

$$= \min_{\hat{\mu}, \hat{\Lambda}} \int_{\mathcal{X}} p(x) \frac{1}{2} (\hat{\Lambda}(x) \hat{r}(x)^2 - \log \hat{\Lambda}(x)) dx \quad (21)$$

where  $\hat{\Lambda}$  is a constant function and  $\hat{r}(x) = y(x) - \hat{\mu}(x)$ . This provides an upper bound on the integral as we are looking at a restricted class of possible precision functions. Since the precision function is constant the gradient penalty,  $\|\nabla \hat{\Lambda}(x)\|_2^2$ , is zero. There is no penalty on  $\hat{\mu}$  so it can perfectly pass through every data point and the contribution of  $\hat{\Lambda}(x)(\hat{\mu}(x) - y(x))^2$  is zero while  $-\log \hat{\Lambda}(x)$  can become arbitrarily negative. Thus there is no solution if  $\alpha = 0$ .

□

## B EXPERIMENTAL DETAILS

### B.1 DATASETS

We chose 64 datapoints in each of the simulated datasets. The generating processes for each simulated dataset is included in Table 3 and can be seen in Fig. 4. The homoskedastic data is simulated in the same way, but with  $f(x) = 1$ . For testing, we simulate a new dataset of 64 datapoints with the same process. Table 4 summarizes the UCI datasets. We provide a description of the ClimSim climate data in Appendix D.4.

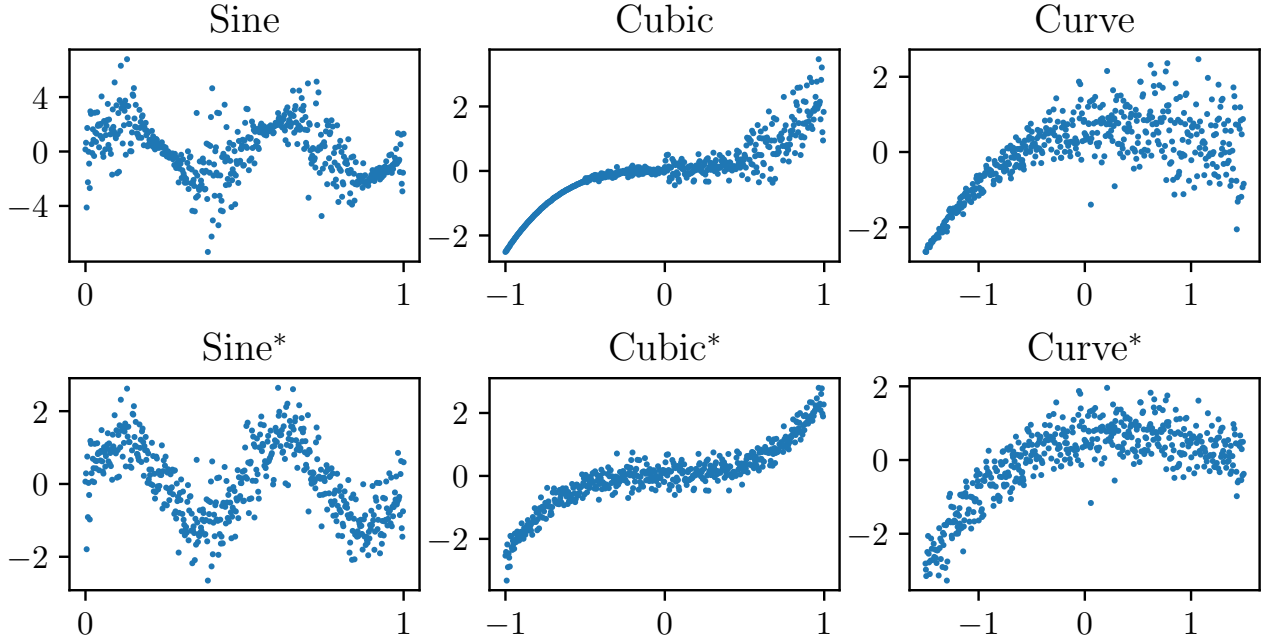


Figure 4: Visualization of heteroskedastic and homoskedastic versions of simulated datasets. Specific details for the functional form of these can be found in Table 3.

Table 3: Simulated datasets. Each dataset is defined by a true  $\mu$  function and then a noise function  $f$ . All data is generated as  $\mu(x) + \epsilon(x)$  where  $\epsilon(x) \sim \mathcal{N}(0, f(x)^2)$ . After the datasets were generated they were scaled to have mean zero and standard deviation one. The homoskedastic versions of each dataset fix  $f(x) = 1$ . The datasets are shown in Fig. 4.

Dataset	Mean ( $\mu$ )	Noise Pattern ( $f$ )	Domain
Sine	$\mu(x) = 2 \sin(4\pi x)$	$f(x) = \sin(6\pi x) + 1.25$	$x \in [0, 1]$
Cubic	$\mu(x) = x^3$	$f(x) = \begin{cases} 0.1 & \text{for } x < -0.5 \\ 1 & \text{for } x \in [-0.5, 0.0) \\ 3 & \text{for } x \in [0.0, 0.5) \\ 10 & \text{for } x \geq .5 \end{cases}$	$x \in [-1, 1]$
Curve	$\mu(x) = x - 2x^2 + 0.5x^3$	$f(x) = x + 1.5$	$x \in [-1.5, 1.5]$

Table 4: UCI dataset.

Dataset	Train Size	Test Size	Input Dimension
Concrete	687	343	8
Housing	337	168	13
Power	6379	3189	4
Yacht	204	102	6

## B.2 TRAINING DETAILS

We take 22 values of  $\gamma, \rho$  that range from  $10^{-10}$  up to  $1 - 10^{-5}$  on a logit scale for all of the experiments run on neural networks. The exact values were  $(\rho, \gamma \in \{0.9999, 0.999, 0.99, 0.9, 0.8, 0.7, 0.6, 0.5, 0.4, 0.3, 0.2, 0.1, 0.01, 0.001, 0.0001, 0.00001, 0.000001, 0.0000001, 0.00000001, 0.000000001, 0.0000000001\})$ . For the FTs we take 20 values from  $10^{-6}$  up to  $1 - 10^{-7}$  also on a logit scale  $(\rho, \gamma \in \{0.999999, 0.99999, 0.99999, 0.9999, 0.999, 0.99, 0.9, 0.8, 0.7, 0.6, 0.5, 0.4, 0.3, 0.2, 0.1, 0.01, 0.001, 0.0001, 0.00001, 0.000001\})$ . This scaling increases the absolute density of points evaluated near the extreme cases of 0 and 1 where the theoretical analysis of the FT focused. The ranges differ slightly due to numerical stability during the fitting. The limiting cases of  $\gamma, \rho \in \{0, 1\}$  were omitted for numerical stability and the ranges of values for the FTs vs neural networks vary slightly for the same reason. The values of  $\rho, \gamma$  that were taken along the  $\rho = 1 - \gamma$  line were  $\rho, \gamma \in \{0.0, 1.0 \times 10^{-11}, 1.0 \times 10^{-10}, 1.0 \times 10^{-9}, 1.0 \times 10^{-8}, 1.0 \times 10^{-7}, 1.0 \times 10^{-6}, 1.0 \times 10^{-5}, 1.0 \times 10^{-4}, 1.0 \times 10^{-3}, 0.01, 0.1, 0.11, 0.12, 0.13, 0.14, 0.15, 0.16, 0.17, 0.18, 0.19, 0.20, 0.21, 0.22, 0.23, 0.24, 0.25, 0.26, 0.27, 0.28, 0.29, 0.30, 0.31, 0.32, 0.33, 0.34, 0.35, 0.36, 0.37, 0.38, 0.39, 0.40, 0.41, 0.42, 0.43, 0.44, 0.45, 0.46, 0.47, 0.48, 0.49, 0.50, 0.51, 0.52, 0.53, 0.54, 0.55, 0.56, 0.57, 0.58, 0.59, 0.60, 0.61, 0.62, 0.63, 0.64, 0.65, 0.66, 0.67, 0.68, 0.69, 0.70, 0.71, 0.72, 0.73, 0.74, 0.75, 0.76, 0.77, 0.78, 0.79, 0.80, 0.81, 0.82, 0.83, 0.84, 0.85, 0.86, 0.87, 0.88, 0.89, 0.9, 0.99, 0.999, 0.9999, 1.0\}$ . All experiments were run on Nvidia Quadro RTX 8000 GPUs. Approximately 500 total GPU hours were used across all experiments.

## B.3 METRICS

- Geometric complexity: For the one-dimensional datasets the function is evaluated on a dense grid and then the gradients are approximated via finite differences and a trapezoidal approximation to the integral is taken. In the case of the FT, we only assess the function on the solved for, discretized points while with the neural networks we interpolate between points. For the higher-dimensional UCI datasets the gradients are also numerically approximated in the same way but only at the points in the train/test sets.
- MSE: In the fully non-parametric, unconstrained setting, the maximum likelihood estimates at each  $x_i$  are  $\hat{\mu}(x_i) = y(x_i)$  and  $\hat{\Lambda}(x_i) = (y(x_i) - \mu(x_i))^{-2} \implies \hat{\Lambda}^{-1/2}(x_i) = |y(x_i) - \mu(x_i)|$ , serving as motivation for checking these differences.

**Variability over runs** The experiments were each run six times with different seeds. The standard deviations over the metrics displayed in Fig. 2 are shown in Fig. 5. The Sobolev norms show that there is the most variability in the overfitting regions  $O_I$  and parts of  $O_{II}$ . This indicates that the functions themselves vary across runs. However, when turning to quality of fits, the MSEs show a different pattern of regions of instability, and  $O_I$  has low variability in terms of actual performance.

## B.4 FIELD THEORY

For the discretized field theory we take  $n_{ft} = 4096$  evenly spaced points on the interval  $[-1, 1]$ . There are two datapoints placed beyond  $[-1, 1]$  because the method we use to estimate the gradients requires the datapoints to have left and right neighbors. These datapoints were not included when computing our metrics. Of these 4096 datapoints 64 were randomly selected to be used for training neural networks  $\hat{\mu}_\theta, \hat{\Lambda}_\phi$ . The field theory results were consistent across choices of  $n_{ft} \in \{256, 512, 1024, 2048, 4096\}$ . We present results for  $n_{ft} = 4096$  in the main paper. We train for 100000 epochs and use the Adam optimizer with a basic triangular cycle that scales initial amplitude by half each cycle on the learning rate. The minimum and maximum learning rates were 0.0005 and 0.01. The cycles were 5000 epochs long. We clip the gradients at 1000. A subset of the fits can be seen in Fig. 10.

## B.5 SIMULATED DATA WITH NEURAL NETWORKS

For all of the simulated datasets except for *Sine* we train for 600000 epochs and use the Adam optimizer with a basic triangular cycle that scales initial amplitude by half each cycle on the learning rate. The minimum and maximum learning rates were 0.0001 and 0.01. The cycles were 50000 epochs. The first 250000 epochs are only spend on training  $\hat{\mu}_\theta$  while the remaining 350000 epochs are spent training both  $\hat{\mu}_\theta, \hat{\Lambda}_\phi$ . We clip the gradients at 1000. The training for the *Sine* dataset was the same, except trained for 2500000 epochs. A subset of the fits for the *Sine* dataset can be seen in Fig. 11.

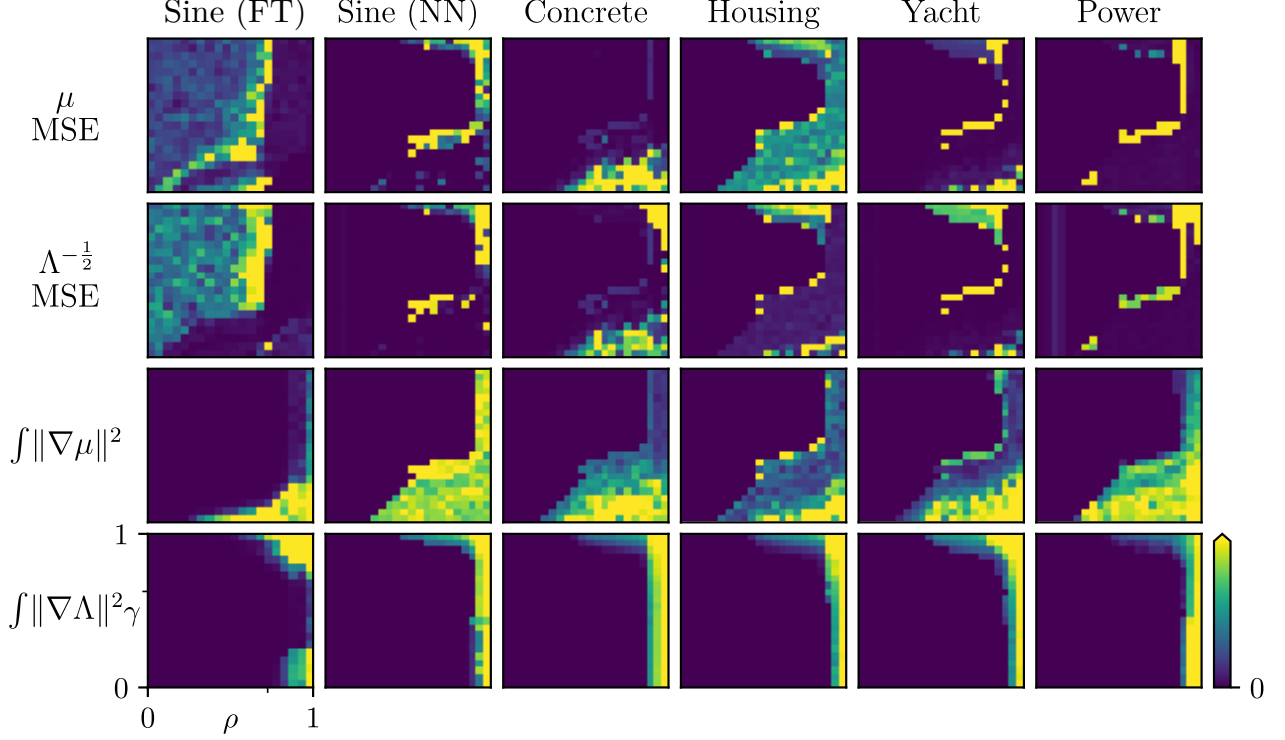


Figure 5: The standard deviation over the six runs of each metric shown in Fig. 2

## B.6 UCI DATA WITH NEURAL NETWORKS

For the *concrete*, *housing* and *yacht* datasets we train for 500000 epochs and use the Adam optimizer with a basic triangular cycle that scales initial amplitude by half each cycle on the learning rate. The minimum and maximum learning rates were 0.0001 and 0.01. The cycles were 50000 epochs. The first 250000 epochs are only spend on training  $\hat{\mu}_\theta$  while the remaining 250000 epochs are spent training both  $\hat{\mu}_\theta, \hat{\Lambda}_\phi$ . Meanwhile on the *power* dataset, we had to use minibatching due to the size of the dataset. We used minibatches of 1000 and trained for 50000 total epochs with the first 25000 dedicated solely to  $\hat{\mu}_\theta$  and the remainder training both  $\hat{\mu}_\theta, \hat{\Lambda}_\phi$ . The same cyclic learning rate was used but with cycle length 5000. We clip the gradients at 1000.

## B.7 PRACTICAL SUGGESTION

We can also view the  $\rho = 1 - \gamma$  line that we search from the perspective of the  $\alpha, \beta$  parameterization of the regularizers. Let  $\rho, \gamma \in (0, 1)$  such that  $\rho = 1 - \gamma$ . Furthermore, we know that  $\alpha = \frac{1-\rho}{\rho}\gamma$  and that  $\beta = \frac{1-\rho}{\rho}(1-\gamma)$ . If we are interested in the model settings for  $(\rho(t) = t, \gamma(t) = 1 - t)$  for  $t \in (0, 1)$ , it then follows that we are equivalently interested in

$$\begin{aligned}
 (\alpha(t), \beta(t)) &= \left( \frac{1-\rho(t)}{\rho(t)}\gamma(t), \frac{1-\rho(t)}{\rho(t)}(1-\gamma(t)) \right) \\
 &= \left( \frac{1-t}{t}(1-t), \frac{1-t}{t}t \right) \\
 &= \left( \frac{(1-t)^2}{t}, 1-t \right) \\
 \implies \begin{cases} t &= 1 - \beta(t) \\ \alpha(t) &= \frac{\beta(t)^2}{t} \end{cases} \text{ or } \sqrt{t\alpha(t)} = \beta(t).
 \end{aligned}$$

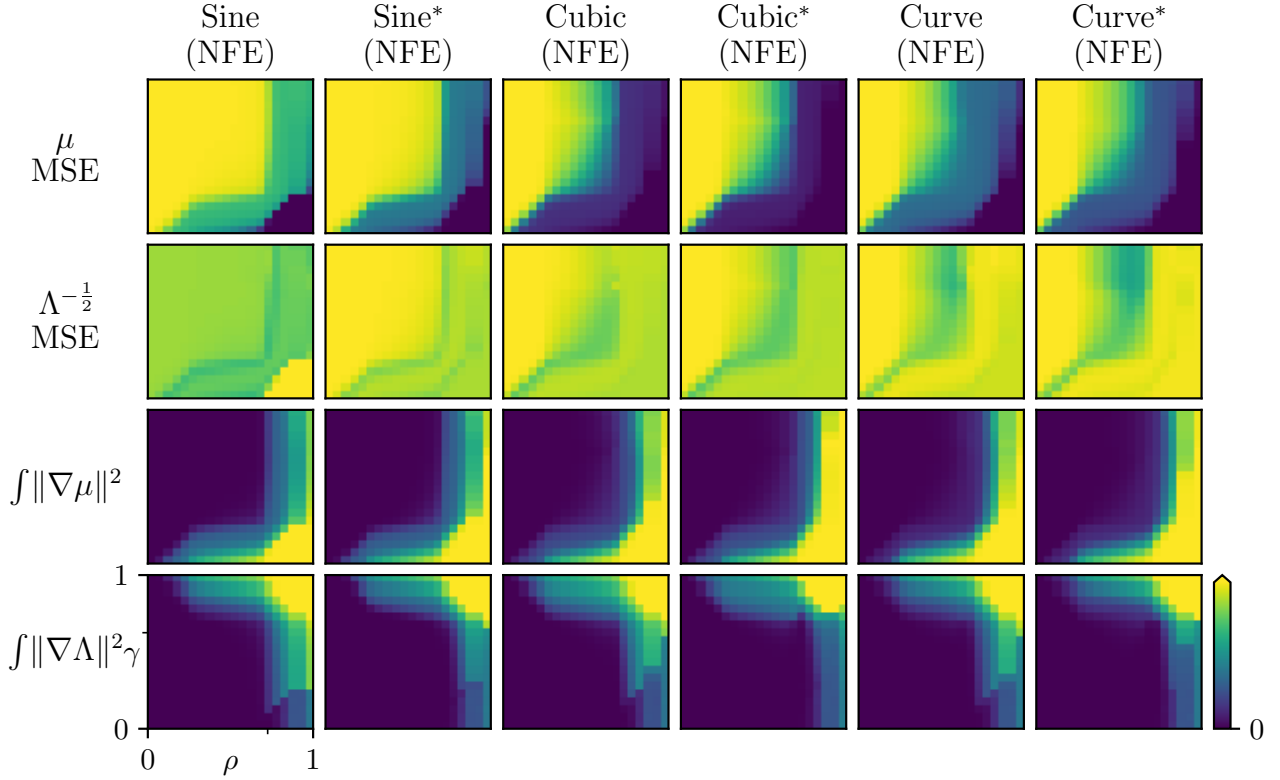


Figure 6: Same configuration as Fig. 2 except all results here pertain to minimizing the FT on six different synthetic datasets described in Table 3. Dataset names with an \* are the homoskedastic counterparts.

## C ADDITIONAL RESULTS

### C.1 ALL SYNTHETIC DATASET RESULTS

Both FT and neural networks were fit to the heteroskedastic and homoskedastic synthetic datasets described in Table 3. The main results for these displayed as phase diagrams of various metrics can be seen in Fig. 6 and Fig. 7 respectively. We largely see the same trends as were exhibited by the real-world datasets seen in Fig. 2.

### C.2 EFFECT OF NEURAL NETWORK SIZE

We used the same training methods to fit models with one and two hidden layers and fit them to the *concrete* dataset. The results in the phase diagram were consistent with the other experiments, as can be seen in Fig. 8.

## D COMPARISON TO BASELINES

We compare the performance of our diagonal  $\rho + \gamma = 1$  search against two baselines,  $\beta$ -NLL [Seitzer et al., 2022] and an ensemble of six MLE-fit heteroskedastic regression models [Lakshminarayanan et al., 2017]. We use  $\mu$  MSE,  $\Lambda^{-\frac{1}{2}}$  MSE, and expected calibration error (ECE) to evaluate the models. In all cases lower values are better. Note that the method of ensembling multiple individual MLE-fit models from [Lakshminarayanan et al., 2017] could be implemented on our method or  $\beta$ -NLL as well.

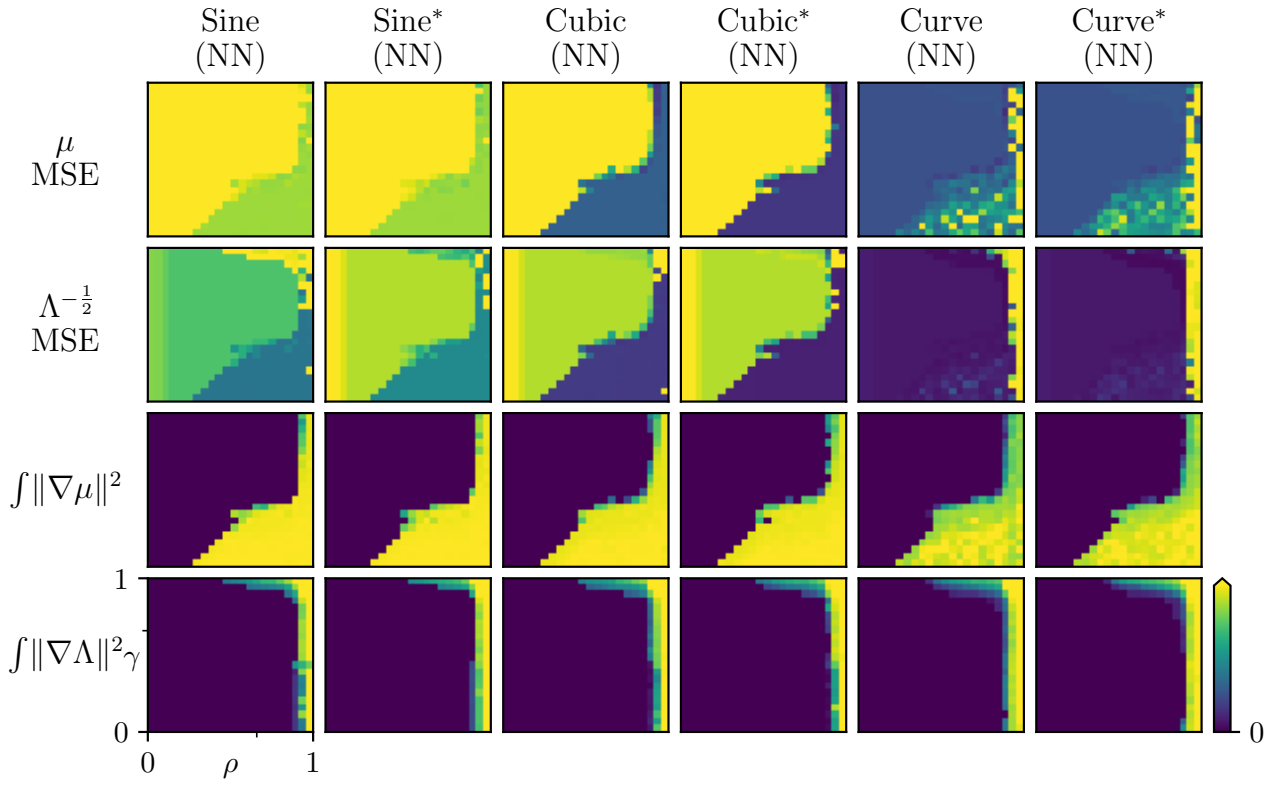


Figure 7: Same configuration as Fig. 2 and Fig. 6, except all results here pertain to training a neural network on six different synthetic datasets described in Table 3. Dataset names with an \* are the homoskedastic counterparts.

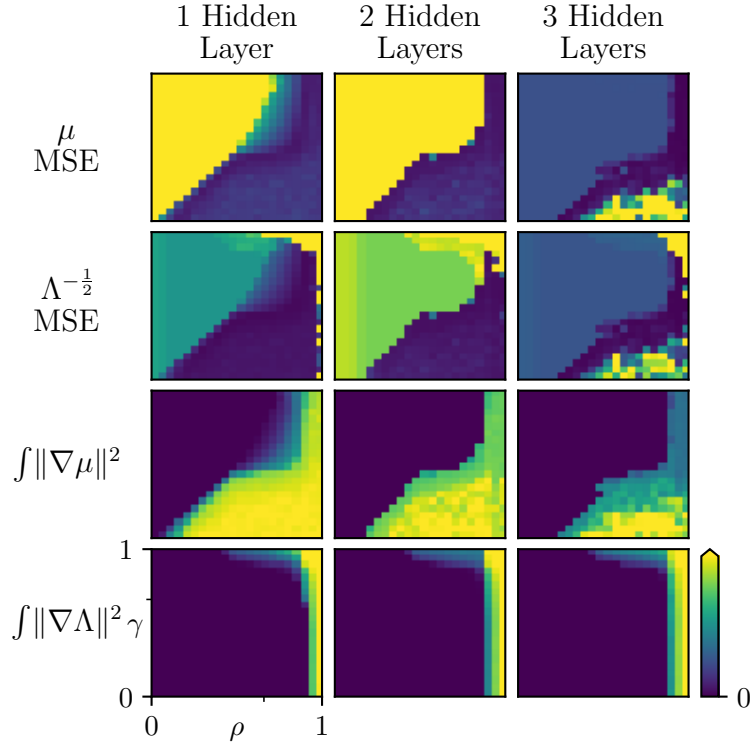


Figure 8: Same configuration as Fig. 2, however, these results pertain all to fitting a neural network of various sizes on the *concrete* dataset.

### D.1 MODEL ARCHITECTURE

All (individual) models have the same architecture: fully connected neural networks with three hidden layers of 128 nodes and leaky ReLU activations for the synthetic and UCI datasets and fully connected neural networks with three hidden layers of 256 nodes for the ClimSim data [Yu et al. 2023]. Note that both baselines model the variance while our approach models the precision (inverse-variance). In all cases we use a softplus on the final layer of the variance/precision networks to ensure the output is positive.

For the  $\beta$ -NLL implementation we take  $\beta = 0.5$  as suggested in [Seitzer et al. 2022]. The ensemble method we use fits 6 individual heteroskedastic neural networks and combines their outputs into a mixture distribution that is approximated with a normal distribution. We do not add in adversarial noise as the authors state it does not make a significant difference. We fit six  $\beta$ -NLL models and six MLE-ensembles.

### D.2 DIAGONAL SELECTION CRITERIA

After conducting our diagonal search we found the model that minimized  $\mu$  MSE and the model that minimized  $\Lambda^{-\frac{1}{2}}$  MSE on the *training* data. In some cases these models coincided. We then used the model that was on the midpoint (on a logit scale) of the  $\rho + \gamma = 1$  line between these two models to compare. The results are reported in Table 5. In all cases our method is competitive with or exceeds the performance of these two baselines—particularly on real-world data. Note that our goal is to show that we are able to find models that model the mean and standard deviation of the data well, that is, lie in our proposed region  $S$  of the phase diagram. We do not claim that this method will always provide optimal model within  $S$ .

### D.3 TRAINING DETAILS

Training for our method was conducted as described in sections B.2 and B.6 of the appendix.

For the baselines, on all of the simulated datasets we train for 600000 epochs and use the Adam optimizer with a basic



triangular cycle that scales initial amplitude by half each cycle on the learning rate. The minimum and maximum learning rates were 0.0001 and 0.01. The cycles were 50000 epochs. We clip the gradients at 1000. The same optimization scheme is performed for the UCI datasets but for 500000 epochs for the *Housing*, *Concrete*, and *Yacht* datasets. The *Power* dataset was trained for 50000 epochs with batches of 1000.

#### D.4 CLIMSIM DATASET

The ClimSim dataset [Yu et al., 2023] is a largescale climate dataset. Its input dimension is 124 and output dimension is 128. We use all 124 inputs to model a single output, *Visible direct solar flux, SOLS* [ $W/m^2$ ]. We train on 10,091,520 of the approximately 100 million points for training and we use a randomly selected 7,209 points to evaluate our models.

#### D.5 DEFICIENCY OF ECE

Shortcomings of ECE (in isolation) are well documented [Kuleshov et al., 2018, Levi et al., 2022, Chung and Neiswanger, 2021]. The main issue with ECE is it measures *average* calibration, while *individual* calibration is more desirable. On our diagonal search we found that often times the models that achieved the best ECE were those that were severely underfit and belonged to region  $U_I$ . In Table 5 we see that the MLE-ensemble is able to achieve low scores while being uncompetitive with respect to the two MSE metrics. The MLE-ensembles were unstable on several of the datasets with respect to the variance network which is consistent with Proposition 1. In particular this can be seen for the synthetic datasets the  $\Lambda^{-\frac{1}{2}}$  MSE diverges to infinity.

### E FOURIER FEATURE MAPPINGS AND GEOMETRIC COMPLEXITY

As a preprocessing step we apply Fourier feature mapping [Tancik et al., 2020] before passing our data into MLPs. That is, we map inputs  $x \rightarrow \gamma(x)$  where  $\gamma(\cdot)$  is defined as follows

$$\gamma(x) = [\cos(2\pi b_1^T x), \dots, \cos(2\pi b_k^T x)]$$

and the  $\{b_i\}_{i=1}^k$  are independently sampled from a  $\mathcal{N}(0, 1)$  distribution. This method has been shown to allow MLPs to learn high frequency data and to reduce training time. The motivation for this additional step is to encourage the mean and precision networks to overfit and hopefully, mimic some of the behaviors of the FT more faithfully. We try this in two different settings. In the first we add in the Fourier Features layers with an L2 penalty and then again with the Dirichlet energy/geometric complexity as the regularizer similar in spirit to [Hoffman et al., 2019]. Just as in the earlier experiments we penalize the mean and precision networks separately and weigh the regularizers in the same  $(\rho, \gamma)$ -scheme.

In the case where we penalize the geometric complexity the regularizer now matches exactly the regularizer of the field theory setting. We find even greater correspondence between results. However, there is a heavy computational burden where training takes on the order of 10 times slower in wall clock time.

We use 2-layer MLPs with width 128 and set the Fourier features mapping to be 64-dimensional, with  $\sigma = 2$  when sampling the weights for both the  $\mu$  and  $\Lambda$  networks. We remove one layer from the MLPs (compared to the earlier experiments) to accommodate the fact that we add in the Fourier feature mapping. We train the models to 128 samples from the generated in the same way as the *Sine* dataset with 5000 epochs warmup for the mean network and 150000 epochs total and have batch size of 32. Despite fewer training epochs than the earlier neural network experiments, we still achieve overfitting behavior. We use the same set of  $(\gamma, \rho)$ -pairings as in the neural network experiments described in section B.5. Summary plots of  $\mu$ - and  $\Lambda^{-1/2}$ -MSE as well as Dirichlet energies can be found in Fig. 9. A subset of resulting fits can be seen in Figs. 12 and 13.

Table 5: Comparison of a deep heteroskedastic regression model with diagonal regularization search against two baselines [Seitzer et al., 2022, Lakshminarayanan et al., 2017]. For details on the selection criteria of the heteroskedastic model see Appendix D.2. We report the average and standard deviations of expected calibration error (ECE),  $\mu$  MSE and  $\Lambda^{-\frac{1}{2}}$  MSE on test data. Lowest mean value for each metric is bolded. In several cases the MLE ensemble failed to properly converge (yielding  $\text{inf} \pm \text{nan}$  results when the standard deviation function diverges to infinity). Individually, there are many pitfalls to using MLE to train heteroskedastic regression models, and it only takes one member of the ensemble to fail to diverge to yield these results. In particular, note that these numerical issues occur most commonly for the quantities relating to the standard deviation. This highlights the instability of MLE training in this setting. Note that the method of ensembling multiple individual MLE-fit models could be performed on our method or  $\beta$ -NLL as well.

Dataset		Heteroskedastic	$\beta$ -NLL	MLE Ensemble
	Metric		Seitzer et al. [2022]	Lakshminarayanan et al. [2017]
Cubic	ECE	<b>0.2380</b> $\pm$ 0.03	0.2385 $\pm$ 0.02	0.2411 $\pm$ 0.02
	$\mu$ MSE	0.2339 $\pm$ 0.01	<b>0.1500</b> $\pm$ 0.01	1.1809 $\pm$ 1.88
	$\Lambda^{-\frac{1}{2}}$ MSE	0.2397 $\pm$ 0.02	<b>0.1397</b> $\pm$ 0.01	inf $\pm$ nan
Curve	ECE	0.1804 $\pm$ 0.02	<b>0.1754</b> $\pm$ 0.02	0.2432 $\pm$ 0.00
	$\mu$ MSE	<b>0.4318</b> $\pm$ 0.12	0.4877 $\pm$ 0.16	1.0067 $\pm$ 0.19
	$\Lambda^{-\frac{1}{2}}$ MSE	0.4655 $\pm$ 0.09	<b>0.4187</b> $\pm$ 0.20	inf $\pm$ nan
Sine	ECE	0.2499 $\pm$ 0.00	<b>0.2082</b> $\pm$ 0.03	0.2313 $\pm$ 0.05
	$\mu$ MSE	<b>0.7968</b> $\pm$ 0.00	4.4107 $\pm$ 6.90	0.9716 $\pm$ 0.06
	$\Lambda^{-\frac{1}{2}}$ MSE	<b>0.7968</b> $\pm$ 0.00	4.3524 $\pm$ 6.89	inf $\pm$ nan
Concrete	ECE	0.2471 $\pm$ 0.01	0.2552 $\pm$ 0.00	<b>0.0655</b> $\pm$ 0.01
	$\mu$ MSE	<b>0.1055</b> $\pm$ 0.02	0.5461 $\pm$ 0.30	2.2454 $\pm$ 1.74
	$\Lambda^{-\frac{1}{2}}$ MSE	<b>0.3028</b> $\pm$ 0.51	1.0867 $\pm$ 0.20	$1.3 \times 10^5 \pm 1.2 \times 10^5$
Housing	ECE	0.0653 $\pm$ 0.00	<b>0.2631</b> $\pm$ 0.01	0.1332 $\pm$ 0.02
	$\mu$ MSE	<b>1.2236</b> $\pm$ 0.00	0.3175 $\pm$ 0.06	155.4494 $\pm$ 128.27
	$\Lambda^{-\frac{1}{2}}$ MSE	<b>0.7610</b> $\pm$ 0.00	0.8820 $\pm$ 0.03	218.8269 $\pm$ 195.38
Power	ECE	0.2233 $\pm$ 0.01	0.2370 $\pm$ 0.00	<b>0.0285</b> $\pm$ 0.01
	$\mu$ MSE	0.0350 $\pm$ 0.01	0.1013 $\pm$ 0.01	<b>0.0177</b> $\pm$ 0.00
	$\Lambda^{-\frac{1}{2}}$ MSE	0.0343 $\pm$ 0.01	0.3081 $\pm$ 0.37	<b>0.0091</b> $\pm$ 0.00
Yacht	ECE	0.3038 $\pm$ 0.04	0.2882 $\pm$ 0.02	<b>0.0463</b> $\pm$ 0.02
	$\mu$ MSE	<b>0.0077</b> $\pm$ 0.01	0.0137 $\pm$ 0.01	6.2670 $\pm$ 13.96
	$\Lambda^{-\frac{1}{2}}$ MSE	<b>0.0076</b> $\pm$ 0.01	1.3275 $\pm$ 0.02	8.0599 $\pm$ 19.18
Solar Flux	ECE	<b>0.1503</b> $\pm$ 0.00	0.3007 $\pm$ 0.00	0.1924 $\pm$ 0.04
	$\mu$ MSE	<b>0.2887</b> $\pm$ 0.00	0.3771 $\pm$ 0.00	1.0067 $\pm$ 0.19
	$\Lambda^{-\frac{1}{2}}$ MSE	<b>0.1175</b> $\pm$ 0.00	0.3217 $\pm$ 0.00	$4.6 \times 10^9 \pm 9.9 \times 10^9$

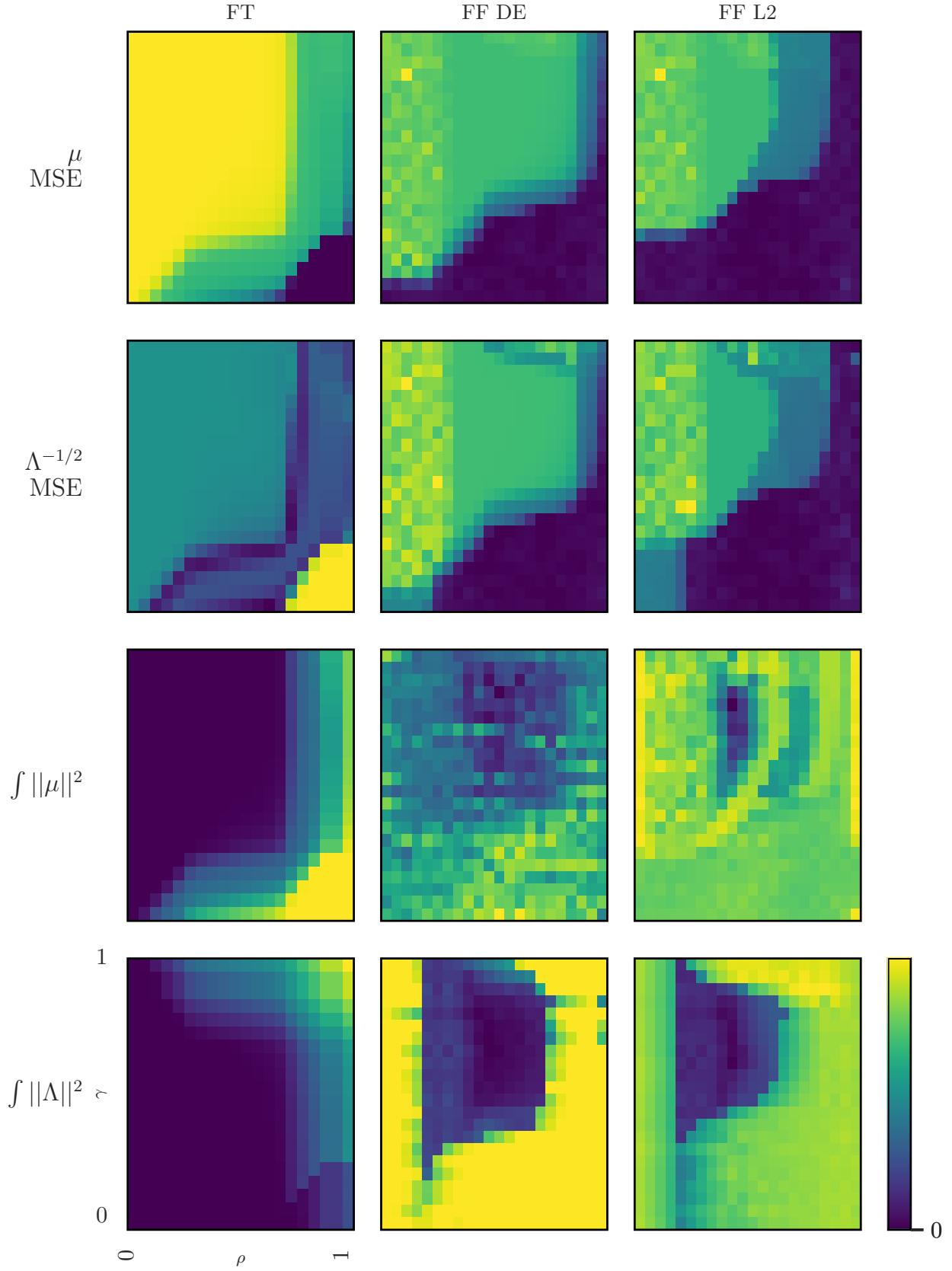


Figure 9: Phase diagrams for the field theory (left), and MLPs with Fourier feature mappings with Dirichlet energy regularization (middle) and L2 regularization (right) fit to the *Sine* dataset.

## Field Theory Fits

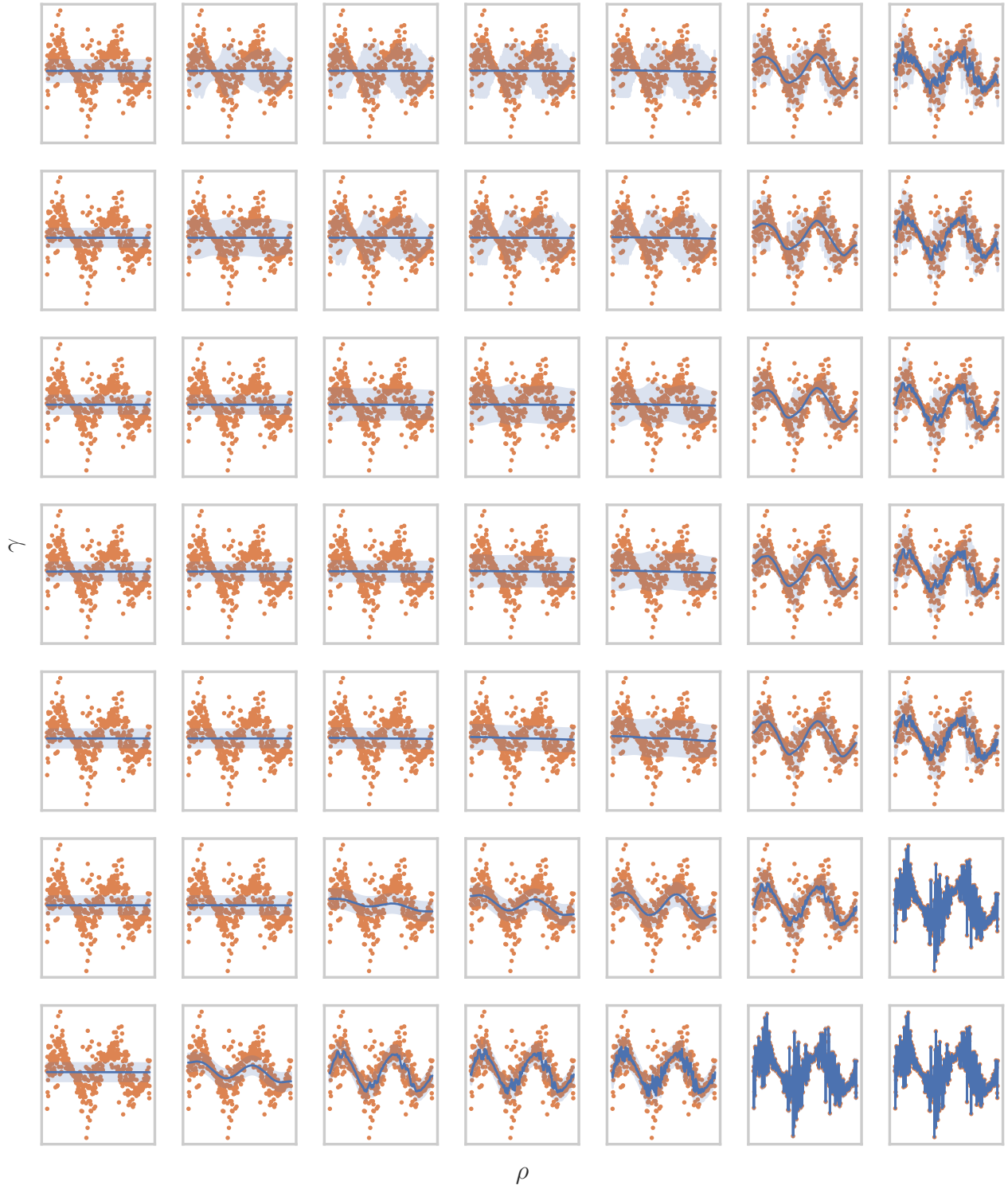


Figure 10: Subsample of fits of the field theory. Moving right to left increases  $\rho$  while moving up and down to up increases  $\gamma$ . Training data is shown in orange, the mean function is shown in red, and  $\pm 1$  SD is shaded. Note: FT was fit to 4096 datapoints, but here we display a thinned subset of the points for visual clarity.

## MLP (L2) Fits

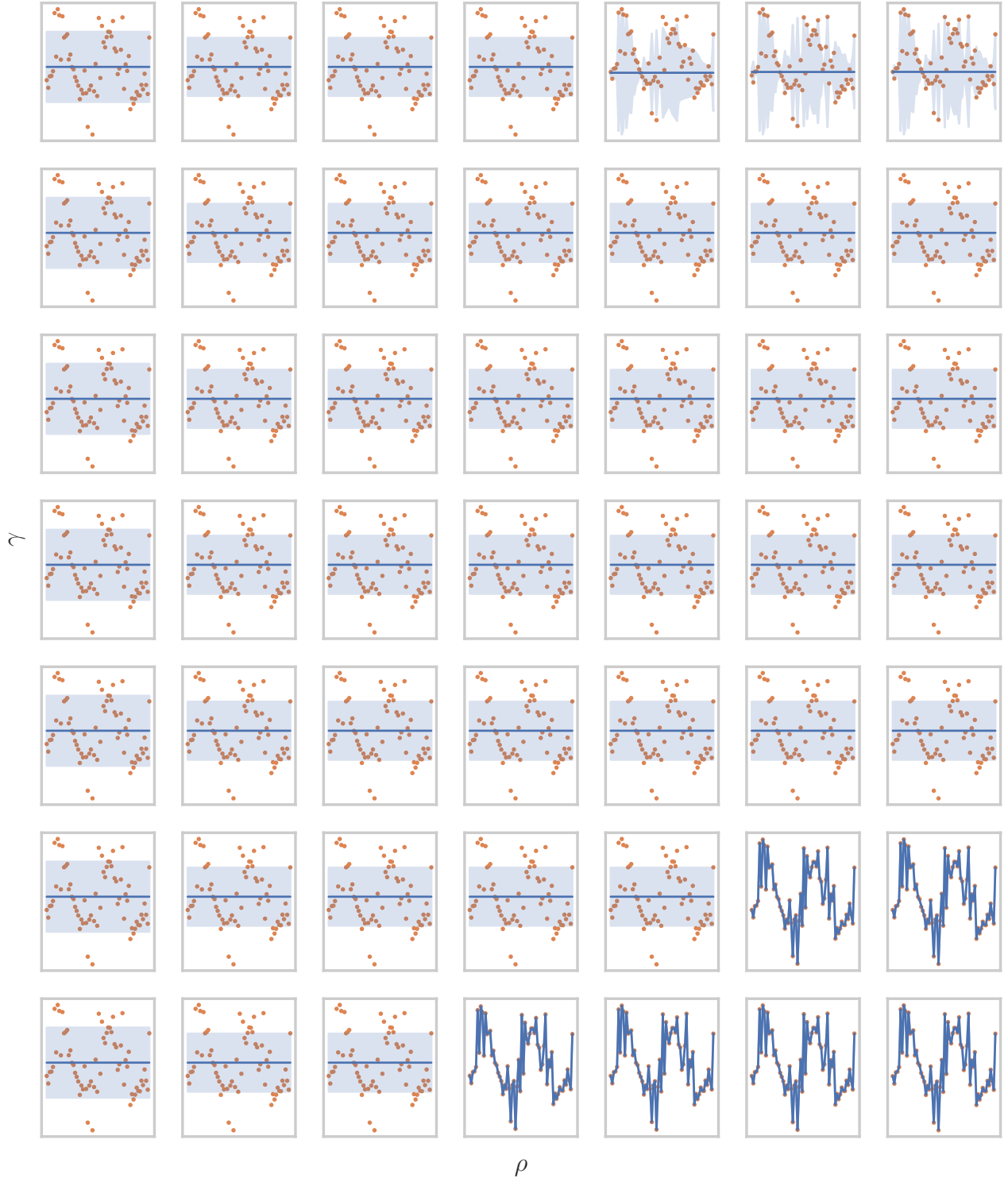


Figure 11: Subsample of fits of the neural networks on the *Sine* dataset. Training data is shown in orange, the mean function is shown in red, and  $\pm 1$  SD is shaded. Notice the abrupt phase transition from overfitting to underfitting in the mean function (in the lower right corner) and similarly in the precision function (in the upper right corner). Moving right to left increases  $\rho$  while moving up and down to up increases  $\gamma$ .

## Fourier Feature (DE) Fits

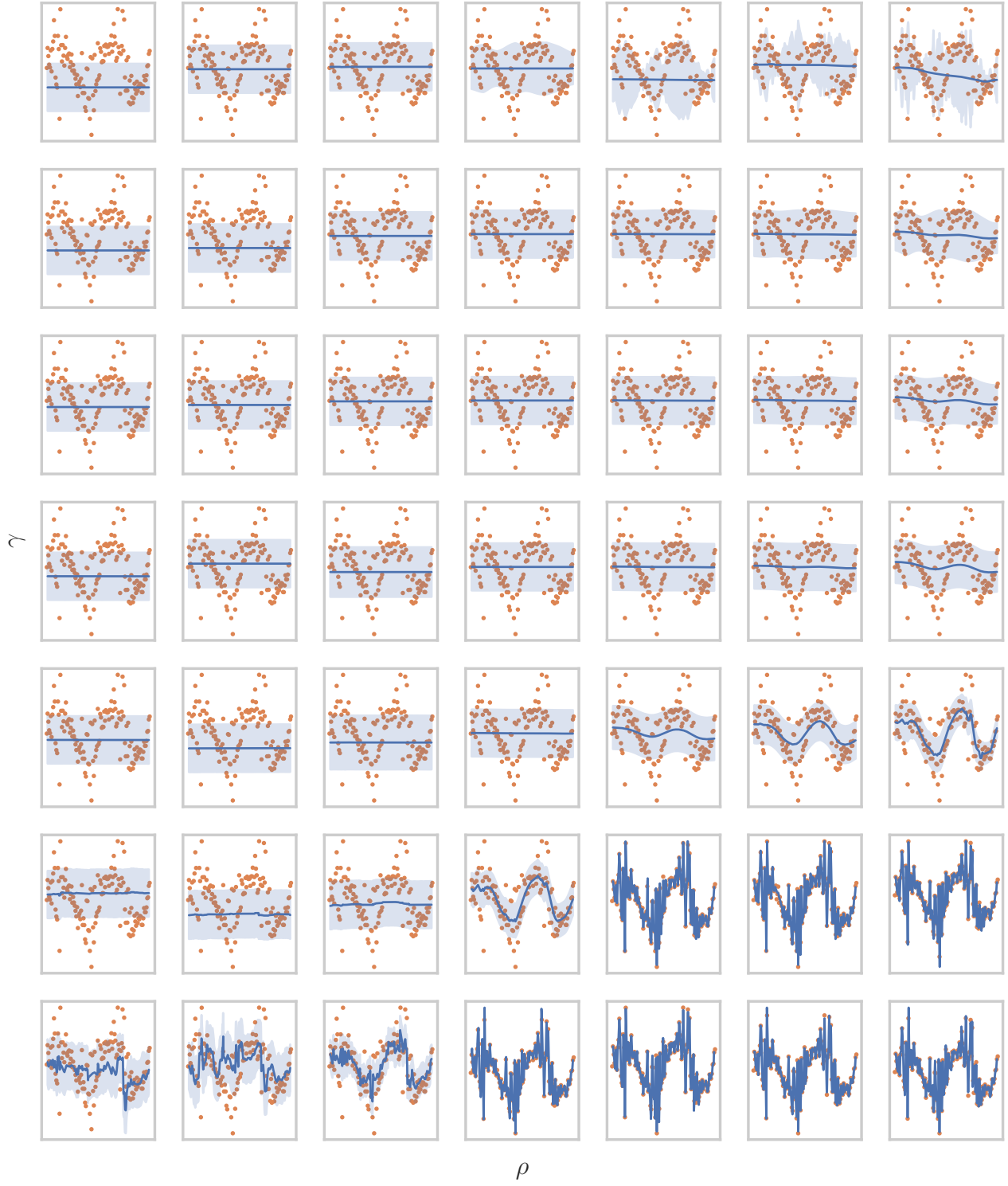


Figure 12: Subsample of fits of neural networks with Fourier Feature layers and Dirichlet energy/geometric complexity regularization. Training data is shown in orange, the mean function is shown in red, and  $\pm 1$  SD is shaded. Moving right to left increases  $\rho$  while moving up and down to up increases  $\gamma$ .

# Fourier Feature (L2) Fits

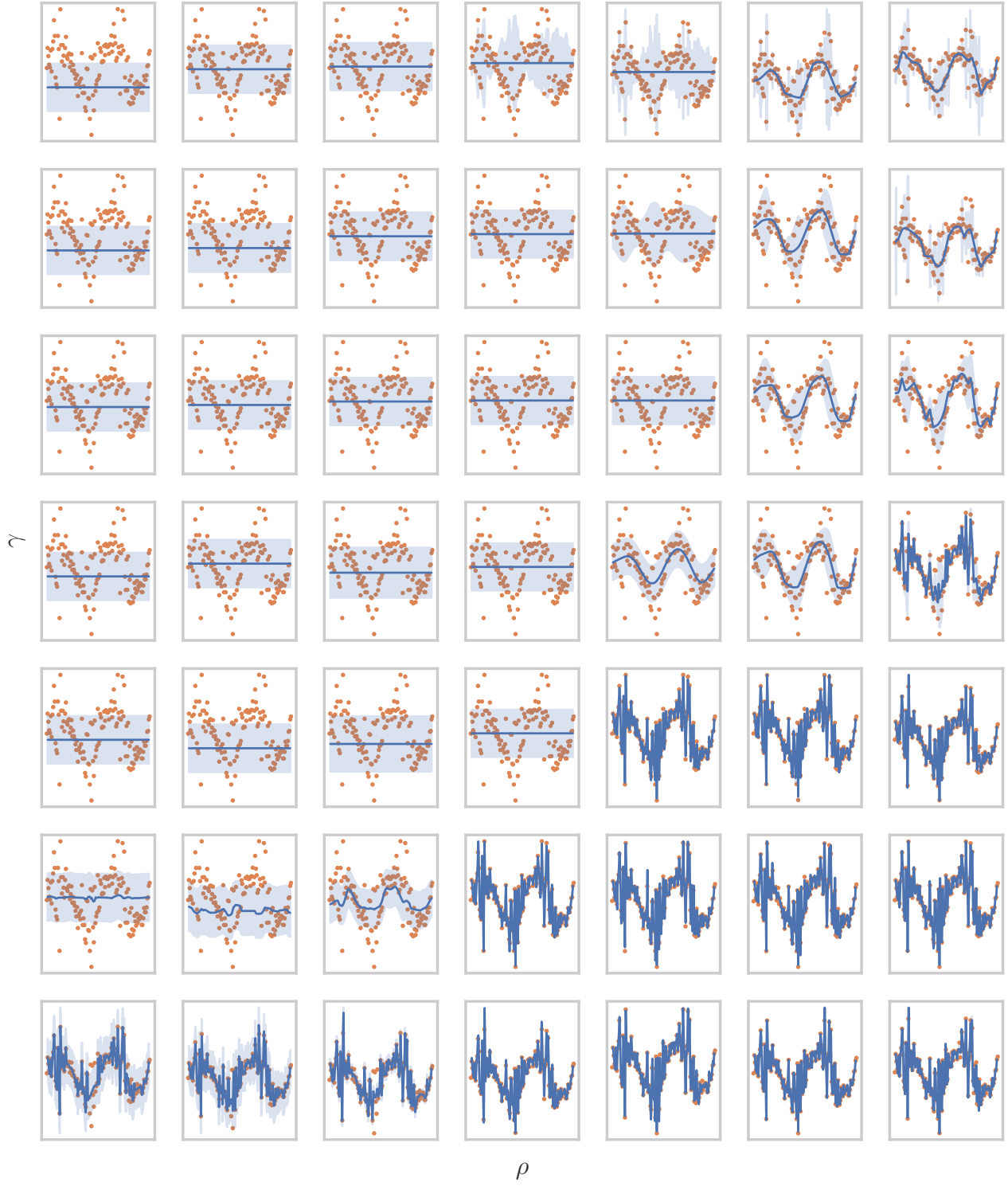


Figure 13: Subsample of fits of neural networks with Fourier Feature layers and L2 complexity regularization. Training data is shown in orange, the mean function is shown in red, and  $\pm 1$  SD is shaded. Moving right to left increases  $\rho$  while moving up and down to up increases  $\gamma$ .