Deep and Multi-fidelity learning with Gaussian processes

Andreas Damianou, Sr ML Scientist

Advances in Data Science Seminar Series, Univ. Manchester
15 Oct. 2019

andreasdamianou.com
Various parts of this talk come from work with:

Neil Lawrence
Kurt Cutajar
Paris Perdikaris
Mark Pullin
Javier Gonzalez
Gaussian processes (GPs):
  • Reasoning about functions through smoothness assumptions

Deep Gaussian processes (DGPs)
  • A much richer class of (deep) models: compositions of GPs

Multi-fidelity modelling with DGPs:
  • Learn from multiple sources by treating the layers of DGP as fidelities
Curve fitting

▸ Which curve fits the data better?
▸ Which curve is more “complex”?
▸ Which curve is better overall?
Curve fitting

- Which curve fits the data better?
- Which curve is more “complex”?
- Which curve is better overall?

Assumptions
- Occam’s razor
Curve fitting

- Which curve fits the data better?
- Which curve is more “complex”?  
- Which curve is better overall?

Assumptions
- Occam’s razor  
- Gaussian process
Posterior probability

- Posterior inference over space of functions

\[ \text{posterior} \propto \text{likelihood} \times \text{prior} \]

Signal from observed data \quad prior assumptions
Part 1: Gaussian processes

See also:

adamian.github.io/talks/Damianou_GP_tutorial.html
Polynomial Regression

Interpolation

Extrapolation

Gaussian Process Regression
Introducing Gaussian Processes:

- A Gaussian distribution depends on a mean and a covariance matrix.
- A Gaussian process depends on a mean and a covariance function.
Sampling from a 2-D Gaussian
Infinite model... but we *always* work with finite sets!

Let’s start with a multivariate Gaussian:

\[
p\left(\{f_1, f_2, \cdots, f_s, f_{s+1}, f_{s+2}, \cdots, f_N\} \mid \{z_A, z_{s+1}, z_{s+2}, \cdots, z_N\}\right) \sim \mathcal{N}(\mu, K).
\]

with:

\[
\mu = \begin{bmatrix} \mu_A \\ \mu_B \end{bmatrix} \quad \text{and} \quad K = \begin{bmatrix} K_{AA} & K_{AB} \\ K_{BA} & K_{BB} \end{bmatrix}
\]

Marginalisation property:

\[
p(f_A, f_B) \sim \mathcal{N}(\mu, K). \quad \text{Then:} \quad p(f_A) = \int_{f_B} p(f_A, f_B)df_B = \mathcal{N}(\mu_A, K_{AA})
\]
Infinite model… but we *always* work with finite sets!

Let’s start with a multivariate Gaussian:

\[
p(\underbrace{f_1, f_2, \cdots, f_s}_{f_A}, \underbrace{f_{s+1}, f_{s+2}, \cdots, f_N}_{f_B}) \sim \mathcal{N}(\mu, K).
\]

with:

\[
\mu = \begin{bmatrix} \mu_A \\ \mu_B \end{bmatrix} \quad \text{and} \quad K = \begin{bmatrix} K_{AA} & K_{AB} \\ K_{BA} & K_{BB} \end{bmatrix}
\]

Marginalisation property:

\[
p(f_A, f_B) \sim \mathcal{N}(\mu, K). \quad \text{Then:}
\]

\[
p(f_A) = \int_{f_B} p(f_A, f_B) df_B = \mathcal{N}(\mu_A, K_{AA})
\]
In the GP context $f = f(x)$:

$$
\mu_{\infty} = \begin{bmatrix} \mu_f \\ \vdots \\ \vdots \\ \vdots \\ \vdots \end{bmatrix} \quad \text{and} \quad K_{\infty} = \begin{bmatrix} K_{ff} & \cdots \\ \cdots & \cdots \end{bmatrix}
$$
Infinite model... but we always work with finite sets!

In the GP context $f = f(x)$:

$$\mu_\infty = \begin{bmatrix} \mu_f \\ \vdots \\ \vdots \end{bmatrix} \quad \text{and} \quad K_\infty = \begin{bmatrix} K_{ff} & \cdots \\ \vdots & \ddots \end{bmatrix}$$

**Covariance function:** Maps locations $x_i, x_j$ of the input domain $\mathcal{X}$ to an entry in the covariance matrix:

$$K_{i,j} = k(x_i, x_j)$$

For all available inputs:

$$K = K_{ff} = k(X, X)$$
GP: joint Gaussian distribution of the training and the (potentially infinite!) test data:

\[ f^* = f(x^*) \]

\[
\begin{bmatrix}
  f \\
  f^*
\end{bmatrix}
\sim \mathcal{N}
\left(
0,
\begin{bmatrix}
  K & K_* \\
  K^\top & K_{*,*}
\end{bmatrix}
\right)
\]
GP: joint Gaussian distribution of the training and the (potentially infinite!) test data:

\[ f^* = f(x^*) \]

\[
\begin{bmatrix} f \\ f^* \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} K & K_* \\ K_\top & K_{**,*} \end{bmatrix} \right)
\]

\( K_* \) is the (cross)-covariance matrix obtained by evaluating the covariance function in pairs of training inputs \( X \) and test inputs \( X_* \), i.e.

\[ f_* = k(X, X_*) . \]

Similarly:

\[ K_{**,*} = k(X_*, X_*) . \]
Posterior is also Gaussian!

\[ p(f_A, f_B) \sim \mathcal{N}(\mu, K). \quad \text{Then:} \]
\[ p(f_A | f_B) = \mathcal{N}(\mu_A + K_{AB} K_{BB}^{-1} (f_B - \mu_B), K_{AA} - K_{AB} K_{BB}^{-1} K_{BA}) \]

In the GP context this can be used for inter/extrapolation:

\[ p(f_* | f_1, \cdots, f_N) = p(f(x_*) | f(x_1), \cdots, f(x_N)) \sim \mathcal{N} \]
\[ p(f_* | f_1, \cdots, f_N) = p(f(x_*) | f(x_1), \cdots, f(x_N)) \sim \mathcal{N}(K_*^\top K^{-1} f_1, \quad K_{*,*} - K_*^\top K^{-1} K_*) \]

\[ p(f(x_*) | f(x_1), \cdots, f(x_N)) \text{ is a posterior process!} \]
Posterior is also Gaussian!

\[ p(f_A, f_B) \sim \mathcal{N}(\mu, K). \quad \text{Then:} \]
\[ p(f_A|f_B) = \mathcal{N}(\mu_A + K_{AB}K_{BB}^{-1}(f_B - \mu_B), K_{AA} - K_{AB}K_{BB}^{-1}K_{BA}) \]

In the GP context this can be used for inter/extrapolation:

\[ p(f_*|f_1, \cdots, f_N) = p(f(x_*)|f(x_1), \cdots, f(x_N)) \sim \mathcal{N} \]
\[ p(f_*|f_1, \cdots, f_N) = p(f(x_*)|f(x_1), \cdots, f(x_N)) \sim \mathcal{N}(K_*^TK^{-1}f, K_{*,*} - K_*^TK^{-1}K_*) \]

\[ p(f(x_*)|f(x_1), \cdots, f(x_N)) \text{ is a posterior process!} \]
Fitting the data \((\text{shaded area is uncertainty})\)
Fitting the data - Prior Samples
Fitting the data
Fitting the data
Fitting the data - Posterior samples
Summary – Gaussian processes

- GPs generalize Gaussian distributions to infinite dimensions (i.e. functions)

- A GP does not have parameters. We only make implicit assumptions about the properties of the functions (e.g. smoothness).

- Predictions are analytic and come with uncertainty.
Part 2: Deep Gaussian processes
A general family of probabilistic models

\[
Y = f_3(f_2(\cdots f_1(X))), \quad H_i = f_i(H_{i-1})
\]
Deep Gaussian process

- Nested function composition
- Non-parametric, non-linear mappings $f$
- Mappings $f$ marginalized out analytically
- NOT a GP!

*Damianou & Lawrence, 2013, Damianou, PhD Thesis 2015*
Sampling from a Deep GP

Input

Unobserved

Output
Regularities are learned as “knots” in the latent space, carried over from layer to layer.
Step function example

Single GP

2-layer Deep GP

4-layer Deep GP
Successive warping to learn the step function

\[ f_1 = f_1(x) \]

\[ f_2 = f_2(f_1) \]

\[ f_3 = f_3(f_2) \]

\[ f_4 = f_4(f_3) \]
Properties

- Unsupervised learning possible due to Bayesian regularization
- Very data efficient
- Scalability also possible with newer techniques

- Intractable objective
- Classification is more challenging
Summary – Deep Gaussian processes

- A DGP is a GP whose input is a GP, whose input is a GP...

- Propagate uncertainty across layers (not only point estimates)
Summary – Deep Gaussian processes

- A DGP is a GP whose input is a GP, whose input is a GP...

- Propagate uncertainty across layers (not only point estimates)

- What if layers represent different kinds of observation spaces? E.g. different fidelities?
Part 3: Multi-fidelity modeling
Multi-fidelity data

High fidelity observations

Low fidelity observations

High fidelity simulations

Low fidelity simulations
Multi-fidelity data

High fidelity observations

High fidelity simulations

Low fidelity observations

Low fidelity simulations
\[ X_H \]
\[ Y_H = \]
OK
OK
OK
ERROR!
OK
OK

\[ X_L \]
\[ Y_L = \]
OK
OK
ERROR!
OK
ERROR!
ERROR!
Fusing information from multiple fidelities

We want to trust the high-fidelity data, where we have them, and where we don’t have them to learn how to reason based on low-fidelity data.
Linear GP multi-fidelity

\[ f_H(x) = \rho_H f_L(x) + \delta_H(x) \]

- \( f_H \): High fidelity function (GP)
- \( \rho_H \): Contribution of low fidelity (const)
- \( f_L \): Low fidelity function (GP)
- \( \delta_H \): Bias between fidelities (GP)

*Kennedy & O’Hagan 2000, Le Gratiet & Garnier 2014*
Non-linear multi-fidelity GP => Deep GP

\[ f_H(x) = \rho_H f_L(x) + \delta_H(x) \]  
Linear relationship between fidelities

\[ f_H(x) = \rho_H (f_L(x), x) + \delta_H(x) \]  
Non-linear relationship between fidelities  
(if \( \rho \) is a GP -&gt; overall a deep GP!)

Non-linear multi-fidelity GP => Deep GP

\[ f_H(x) = \rho_H f_L(x) + \delta_H(x) \quad \text{Linear relationship between fidelities} \]

\[ f_H(x) = \rho_H (f_L(x), x) + \delta_H(x) \]

\[ \downarrow \]

\[ f_H(x) = g_H(f^*_L(x), x) \quad \text{Non-linear relationship between fidelities (if } \rho \text{ is a GP} \implies \text{overall a deep GP!)} \]

\[ \delta \text{ absorbed into } g \]

\[ f^*_L(x) \] denotes the posterior of the GP modeling the low-fidelity data.
1. Train $f_L$ on $(X_L, Y_L)$
2. Compute $f_L^*(X_H)$
3. Train $f_H$ on $((X_L, Y_L), f_L^*(X_H))$
\[ p(f^*_{H}(x^*)) = \int p(f_{H}(x^*, f^*_{L}(x^*)) | y_{H}, x_{H}, x^*) \ p(f^*_{L}(x^*)) \ df^*_{L} \]

- Local posterior from fidelity \(H\)
- Predictive from fidelity \(L\)
Communication between fidelities during training

\[ f_{\text{fidelity layer}} \]

\[ X^1 \]
\[ f_1^{1,2,3} \]
\[ y^1 \]

\[ X^2 \]
\[ f_2^{2,3} \]
\[ y^2 \]

\[ X^3 \]
\[ f_3^3 \]
\[ y^3 \]

A. Damianou

Cutajar et al. arXiv: 1903.07320

10/14/19
Figure 5: Real-world experiment indicating the infection rate of *Plasmodium falciparum* among African children. Lighter-shaded regions denote higher infection rates in that area of the continent. *Left:* True infection rates recorded for the year 2015. *Center:* MF-DGP predictions given low-fidelity data from 2005 and limited high-fidelity training points (marked in red) from 2015. *Right:* White squares show the samples drawn from a DPP using the posterior covariance of the MF-DGP model as its kernel.
Sequential design for multi-fidelity GP

- **Strategy for collecting data points** across fidelities

- Each fidelity evaluation comes with a different **cost**, proportional to the level of fidelity

- Approach: Formulation as a multi-fidelity **bandit** GP problem in the UCB setting (*Kandasamy et al. 2016*)
Demonstration

- Find the maximum of the high-fidelity function
- Consider cost while collecting points from each fidelity
New point: X=[[ 1.]]
New point: $X = [0.76767677]$
Conclusion

- GPs: Non-parametric inference over the space of functions
- Deep GPs learn rich mappings in a regularized and data efficient manner
- Multi-fidelity (D)GPs allow us to fuse multiple fidelity data
- We can actively acquire data of different fidelities using the GP emulators in each fidelity.
Thanks!

Questions?

See also: http://adamian.github.io/talks/Damianou_GP_tutorial.html
Appendix
Unsupervised learning for multiple views

private space

shared space

private space

sample

$Y^{(1)}$

$Y^{(2)}$
Deep Gaussian process

- Objective: $p(y|x) = \int_{h_2} \left( p(y|h_2) \int_{h_1} p(h_2|h_1) p(h_1|x) \right)$

- $p(h_2|x) = \int_{h_1, f_2} p(h_2|f_2) p(f_2|h_1) p(h_1|x)$
Inference in Deep GPs: uncertainty propagation

- Objective: \[ p(y|x) = \int_{h_2} \left( p(y|h_2) \int_{h_1} p(h_2|h_1)p(h_1|x) \right) \]

- \[ p(h_2|x) = \int_{h_1,f_2} p(h_2|f_2) \frac{p(f_2|h_1)}{p(h_1|x)} \]

contains \( (k(h_1,h_1))^{-1} \)

Propagating uncertainty through non-linearities:

\[ y = f(x) + \epsilon \]
Bound on the log marginal likelihood $\log p(y)$

$$
\mathcal{F} = \sum_{l=2}^{L+1} \left\langle \sum_{n=1}^{N} \mathcal{L}(h_l^{(n)}, u_l) \right\rangle_Q - \sum_{l=2}^{L+1} KL (q(u_l) \| p(u_l)) - KL (q(h_1) \| p(h_1)) + \sum_{l=2}^{L} \mathcal{H}(q(h_l))
$$

Data fit

Regularization
Figure 3: Synthetic examples. *Top*: Linear mapping between fidelities. *Bottom*: Nonlinear mapping.
Figure 4: Cross-comparison across methods and synthetic examples for challenging multi-fidelity scenarios. MF-DGP yields conservative uncertainty estimates where few high-fidelity observations are available.
**Benchmark examples**

Table 1: Model Comparison on Multi-fidelity Benchmark Examples.

<table>
<thead>
<tr>
<th>BENCHMARK</th>
<th>$D_{in}$</th>
<th>FIDELITY ALLOCATION</th>
<th>$R^2$</th>
<th>RMSE</th>
<th>MNLL</th>
<th>$R^2$</th>
<th>RMSE</th>
<th>MNLL</th>
<th>$R^2$</th>
<th>RMSE</th>
<th>MNLL</th>
</tr>
</thead>
<tbody>
<tr>
<td>CURRIN</td>
<td>2</td>
<td>12-5</td>
<td>0.913</td>
<td>0.677</td>
<td>20.105</td>
<td>0.903</td>
<td>0.740</td>
<td>20.817</td>
<td><strong>0.935</strong></td>
<td><strong>0.601</strong></td>
<td>0.763</td>
</tr>
<tr>
<td>PARK</td>
<td>4</td>
<td>30-5</td>
<td><strong>0.985</strong></td>
<td>0.575</td>
<td>465.377</td>
<td>0.954</td>
<td>0.928</td>
<td>743.119</td>
<td><strong>0.985</strong></td>
<td><strong>0.565</strong></td>
<td>1.383</td>
</tr>
<tr>
<td>BOREHOLE</td>
<td>8</td>
<td>60-5</td>
<td><strong>1.000</strong></td>
<td><strong>0.005</strong></td>
<td><strong>-3.946</strong></td>
<td>0.973</td>
<td>0.063</td>
<td>-1.054</td>
<td>0.999</td>
<td>0.015</td>
<td>-2.031</td>
</tr>
<tr>
<td>BRANIN</td>
<td>2</td>
<td>80-30-10</td>
<td>0.891</td>
<td>0.044</td>
<td>-1.740</td>
<td>0.929</td>
<td>0.053</td>
<td>-1.223</td>
<td><strong>0.965</strong></td>
<td><strong>0.030</strong></td>
<td><strong>-2.572</strong></td>
</tr>
<tr>
<td>HARTMANN-3D</td>
<td>3</td>
<td>80-40-20</td>
<td><strong>0.998</strong></td>
<td><strong>0.043</strong></td>
<td><strong>0.440</strong></td>
<td>0.305</td>
<td>0.755</td>
<td>0.637</td>
<td>0.994</td>
<td>0.075</td>
<td><strong>-0.731</strong></td>
</tr>
</tbody>
</table>