

Gaussian process regression

ISTA 410 / INFO 510: Bayesian Modeling and Inference

U. of Arizona School of Information

April 19, 2021

Last time:

- Covariance that varies with space or time
- Gaussian processes

Today:

- More on GP regression and covariance kernels
- PyMC3's gp submodule
- Additive GPs

Note: no meeting Wednesday (reading day)

Recap: Gaussian processes

Recap

Last time:

- Model bike share use as a function of weather conditions
 - temperature, windspeed; expect positive temperature effect, negative wind effect
 - modeling complicated by the fact that overall ridership increases ~~as the~~ with time
- Use multivariate normals to extend the usual varying intercepts strategy
 - Draw the entire set of intercepts as a single multivariate normal
 - Impose a covariance structure that makes nearby intercepts highly correlated, distant ones uncorrelated

We tried two versions:

- Intercepts vary monthly, 24 dimensional multivariate normal
- Intercepts vary weekly, 105 dimensional multivariate normal

Both reflect the idea that:

- The intercept is a *function* that varies over time, and we observe the function at various intervals (weekly, monthly)

Function fitting

Taking this to its logical extreme:

- observe this function at increasingly more points, get an infinite-dimensional multivariate normal
- what is an infinite-dimensional multivariate normal? A Gaussian process
 - a multivariate normal is a probability distribution on a space of vectors
 - a Gaussian process is a probability distribution on a space of functions

$$\mu_i = \alpha + \beta x_i \leftarrow \text{linear model}$$

$$\mu(x) \sim \text{GP}(k(x, x')) \leftarrow \text{GP model}$$

Specifying a GP

- A MVNormal is specified by a mean vector and a covariance matrix

$$\vec{y} \sim \text{MVNormal}(\vec{\mu}, \Sigma) \rightarrow p(\vec{y}) =$$

- A GP is specified by a mean function and a covariance function

$$f(x) \sim \mathcal{GP}(\mu(x), k(x, x'))$$

evaluate this "kernel" to get covariance matrix.

- What this means: any finite collection of function values $f(x_1), f(x_2), \dots, f(x_n)$ has a MVNormal distribution:

$$f(\vec{x}) \sim \text{MVNormal}(\mu(\vec{x}), K(\vec{x}))$$

where

$$K(\vec{x})_{ij} = k(x_i, x_j)$$

defining property of a GP

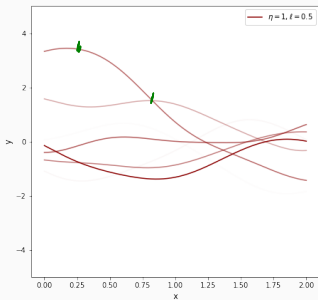
Choosing a covariance function

All of the interesting features of a GP come down to the covariance function:

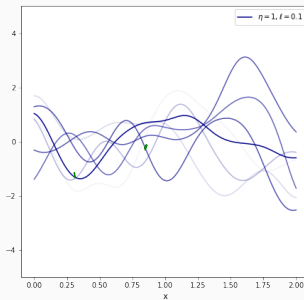
- Common to leave the mean function as 0
- Covariance function influences various properties of the resulting function
 - Smoothness: related to how rapidly correlations decay over short distance
 - Length scale: related to how rapidly correlations decay over longer distances
 - Periodicity: put a periodic component in the correlation function
 - Linear or other trends: covariance that varies with distance from a fixed point

Draws from a prior

Samples from the GP prior: exponentiated quadratic covariance



$GP(k_1)$



$GP(k_2)$

Smoothness

Last time: exponentiated quadratic vs exponential

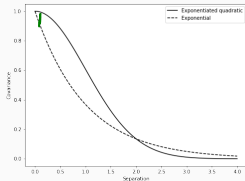
- Exp. Quad:

$$\left\{ \begin{array}{l} k(x, x') = \eta^2 \exp \left(-\frac{|x - x'|^2}{2\ell^2} \right) \end{array} \right.$$

length scale

- Exponential:

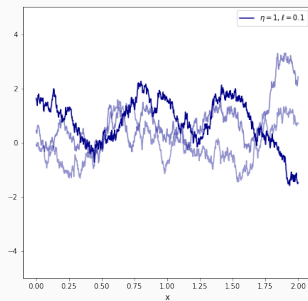
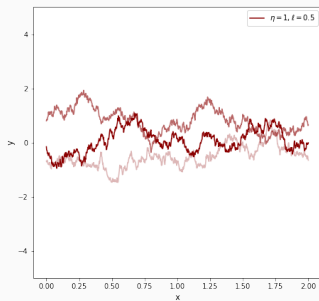
$$k(x, x') = \eta^2 \exp \left(-\frac{|x - x'|}{2\ell^2} \right)$$



- Key difference: correlation for infinitesimally separated x, x'

Smoothness

Samples from the GP prior: exponential covariance



Matern kernels

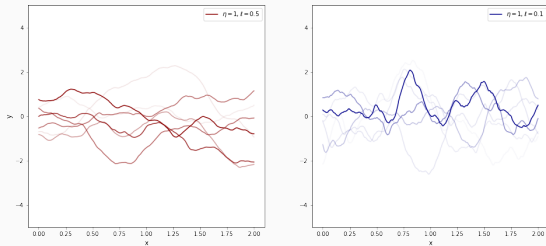
Smoothness interpolation:

- Matern $_{\nu}$ kernel – $\nu = \boxed{1/2}, \boxed{3/2}, \boxed{5/2}, \dots$
 - Formula built with gamma and Bessel functions
- $\nu = 1/2$ equivalent to exponential kernel
- $\nu \rightarrow \infty$ recovers exponentiated quadratic
- Draws from the prior have $\text{floor}(\nu)$ derivatives

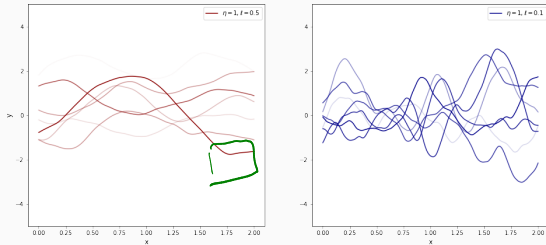
half-integer values

Matern kernels

Samples from the GP prior: Matern(3/2) covariance



Samples from the GP prior: Matern(5/2) covariance



Long term correlations

All of the previous kernels only define “short term” correlations – all go to 0 with distance

- Periodic kernel:

$$k(x, x') = \underbrace{\eta^2}_{\text{magnitude}} \exp \left(- \frac{\sin^2(\pi |x - x'| / \underbrace{T}_{\text{period}})}{2 \underbrace{\ell^2}_{\text{length scale}}} \right)$$

David Mackay

- Linear kernel:

$$k(x, x') = (x - c)(x' - c)$$

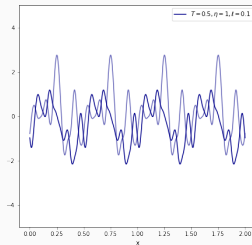
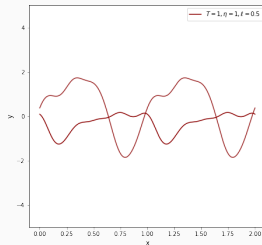
- Polynomial kernel:

$$k(x, x') = [(x - c)(x' - c) + \text{offset}]^d$$

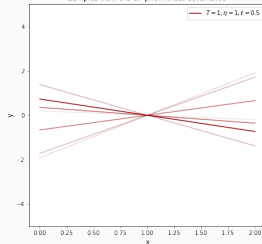
Periodic

Samples from the GP prior: periodic covariance

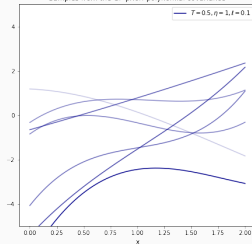
periodic



Samples from the GP prior: linear covariance



Samples from the GP prior: polynomial covariance



PyMC3's gp submodule

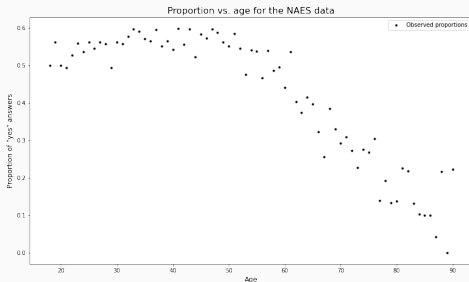
In PyMC3:

- Two implementations of Gaussian processes:
 - Marginal: a little simpler to use — *applicable with normal likelihood*
 - Latent: a little more flexible *↪ applicable with other likelihoods*
- Many standard covariance functions:
 - Exponential, ExpQuad, Matern32, Matern52
 - Periodic, Linear

Example

Exercise 21 from BDA3: data from the 2004 National Annenberg Election Survey

- % of respondents who believe they know someone gay vs. age



Mathematical model

We can write the following model:

replaces
model
equation

$$y(x) \sim \text{Normal}(\mu(x), \sigma)$$

————→ $\mu(x) \sim \mathcal{GP}(0, k)$

$$k(x, x') = \text{Matern}_{5/2}(x, x')$$

magnitude → $\eta \sim \text{Exponential}(1)$

length scale → $\ell \sim \text{Exponential}(0.2)$

$$\sigma \sim \text{HalfCauchy}(1)$$

$$y_i \sim \text{Normal}(\mu_i, \sigma)$$

$$\mu_i = \alpha + \beta x_i$$

$$\alpha \sim \text{Normal}()$$

$$\beta \sim \text{Normal}()$$

$$\sigma \sim \text{HalfCauchy}$$

Notice similarity to a linear regression model!

PyMC3 model

```
eta = pm.Exponential('eta', 1)
ls = pm.Exponential('ls', 0.2)
sigma = pm.HalfCauchy('sigma', 1)
```

} priors

```
cov = pm.gp.cov.Matern52(1, ls=ls)
gp = pm.gp.Marginal(cov_func=cov) # No variable name!
```

(because normal likelihood)

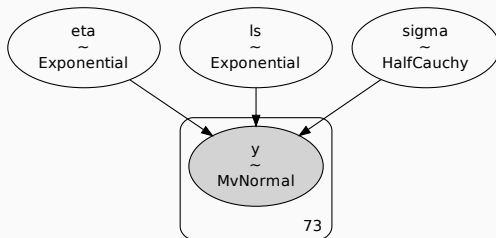
```
y_ = gp.marginal_likelihood('y',
                             X=age, y=prop, noise=sigma)
```

} where model variables
actually get created.

noise can be a covariance function too!

Example

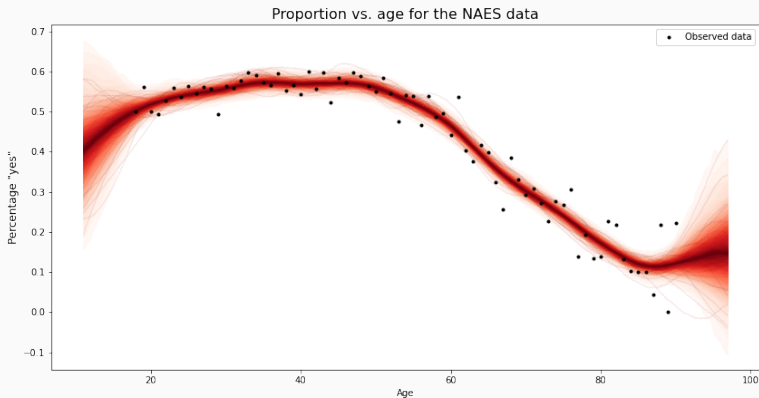
What does the plate diagram look like?



- There's no special Gaussian process variable here – it's just a multivariate normal
- `Marginal` and `marginal_likelihood` are bookkeeping tools

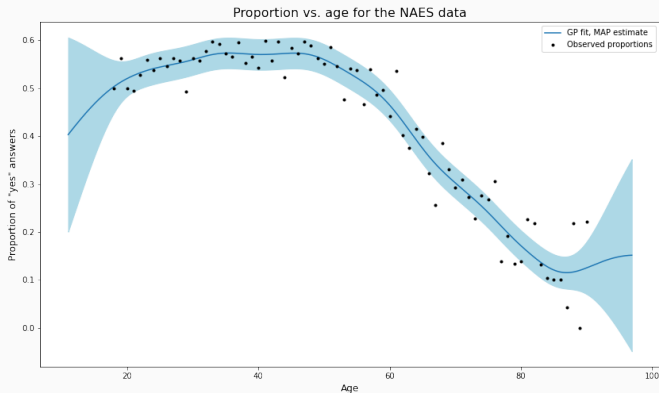
Example

Plotting many samples from the posterior:



Example

Alternative: estimate MAP and use `gp.predict`



Rewriting the bicycle example

- No Gaussian likelihood – Poisson GLM
- Have to use the Latent class here
 - Define the GP with a given mean and covariance function
 - Use the `prior` method to create a multivariate normal for a set of x values

Rewriting the bicycle example

Original version:

```
# Priors for parameters
beta_temp = pm.Normal('beta_temp', 0, 2)
beta_wind = pm.Normal('beta_wind', 0, 2)
eta = pm.Exponential('eta', 1)
ls = pm.Exponential('ls', 1)

# Gaussian process for time-varying intercepts.
cov_matrix = (eta ** 2) * pm.math.exp(-(weekly_distance ** 2)
                                         / (2 * ls ** 2))
                                         + 0.01 * np.eye(105)
alpha = pm.MvNormal('alpha', mu=tt.zeros(105),
                    cov=cov_matrix, shape = 105)

# Model equation, likelihood, sampling
theta = pm.math.exp(alpha[bikes['week']] + beta_temp * bikes['temp']
                    + beta_wind * bikes['windspeed'])
y_ = pm.Poisson('y', theta, observed = bikes['cnt'])
trace = pm.sample(init = 'advi')
```

$$\begin{pmatrix} 0 & 1 & 2 & 3 & \dots \\ 1 & 0 & 1 & 2 & \dots \\ 2 & 1 & 0 & 1 & \dots \\ 3 & 2 & 1 & 0 & \dots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix}$$

replace
with
pm.gp
stuff

← noise in GP

Rewriting the bicycle example

With PyMC3 GP module:

```
# Priors for parameters
beta_temp = pm.Normal('beta_temp', 0, 2)
beta_wind = pm.Normal('beta_wind', 0, 2)
eta = pm.Exponential('eta', 1)
ls = pm.Exponential('ls', 1)
# Gaussian process for time-varying intercepts.
cov_func = (eta ** 2) * pm.gp.cov.ExpQuad(1, ls=ls)
            + 0.01 * pm.gp.cov.WhiteNoise(1)
alpha_gp = pm.gp.Latent(cov_func=cov_func)
f = alpha_gp.prior('f', X=bikes['week'][:, :7, None],
                  , reparameterize=False)
# Model equation, likelihood, sampling
theta = pm.math.exp(f[bikes['week']] + beta_temp * bikes['temp']
                    + beta_wind * bikes['windspeed'])
y_ = pm.Poisson('y', theta, observed = bikes['cnt'])
trace = pm.sample(init = 'advi')
```

define covariance function

role of σ

creates MVN node

take one point from each week

replaces α in model equation

useful for improving sampling performance

Computational issues

Computational difficulties:

- In principle inference is “just” linear algebra if the likelihood is Gaussian
- The linear algebra involves inverting the covariance matrix for the data: $\mathcal{O}(n^3)$
 - Without approximate methods, becomes intractable past a few thousand data points
 - “Cut off” covariance kernels, try to get sparsity in the matrix
 - Compute the covariance matrix only at $m < n$ “inducing points”, reduce to $\mathcal{O}(nm^2)$
- Replace MCMC with MAP estimation or approximate inference methods

} what we
did with
bike share

Additive kernel example

Example: cherry blossom data

Cherry blossoms: big in Japan

- Recorded date of peak flowering
- Records go back to early 10th century CE
- Target: estimate mean peak flowering date over time, separate into slow and fast trends



What might we expect to detect?

Try to detect:

- A long term smooth trend, capturing multi-century trends
- Short-term periodic effects, allowed to decay away from exact periodicity
- Slow trend: exponentiated quadratic kernel
- Faster trend: periodic kernel times Matern(5/2)

$$y_i \sim \text{Normal}(\mu(t), \sigma)$$

$$\mu(t) = f_1(t) + f_2(t)$$

$$f_1(t) \sim \mathcal{GP}(0, k_{\text{slow}})$$

$$f_2(t) \sim \mathcal{GP}(0, k_{\text{fast}})$$

$$k_{\text{slow}}(t, t') = \text{ExpQuad}(t, t')$$

$$k_{\text{fast}}(t, t') = \text{Matern}_{5/2}(t, t') \times \text{Periodic}(t, t')$$

(priors for covariance parameters clipped)

Normal likelihood

inconf function decomposed
into 2

In PyMC3

Periodic component

```
eta_periodic = pm.Exponential('eta_periodic', 0.5)
l_periodic = pm.Gamma('l_periodic', 5, 1/25)
l_periodic_decay = pm.Gamma('l_periodic_decay', 5, 1/25)
period = pm.Exponential('period', 0.001)
periodic_cov = (eta_periodic ** 2)
                * pm.gp.cov.Periodic(1, ls=l_periodic, period = period)
                * pm.gp.cov.ExpQuad(1, ls=l_periodic_decay)
```

```
gp_periodic = pm.gp.Marginal(cov_func=periodic_cov)✱
```

Slow trend

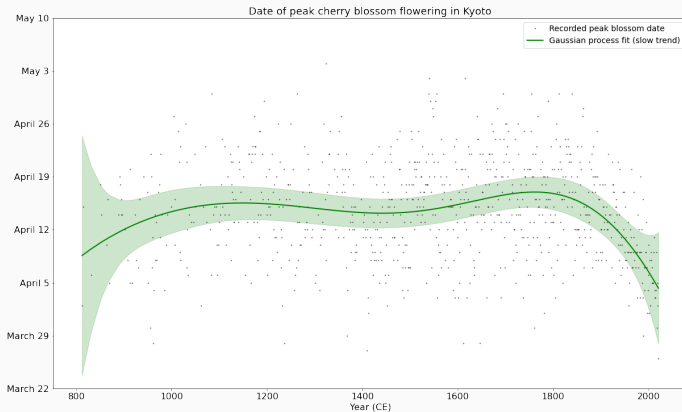
```
eta_slow = pm.HalfCauchy('eta_slow', 2)
l_slow = pm.Gamma('l_slow', 5, 1/25)
slow_cov = (eta_slow ** 2) * pm.gp.cov.ExpQuad(1, ls=l_slow)
gp_slow = pm.gp.Marginal(cov_func=slow_cov)✱
```

#Build model

```
✱ gp_full = gp_slow + gp_periodic
sigma = pm.HalfCauchy('eta_noise', 5)
y_ = gp_full.marginal_likelihood('y', X=t, y=flower_date, noise=sigma) |
```

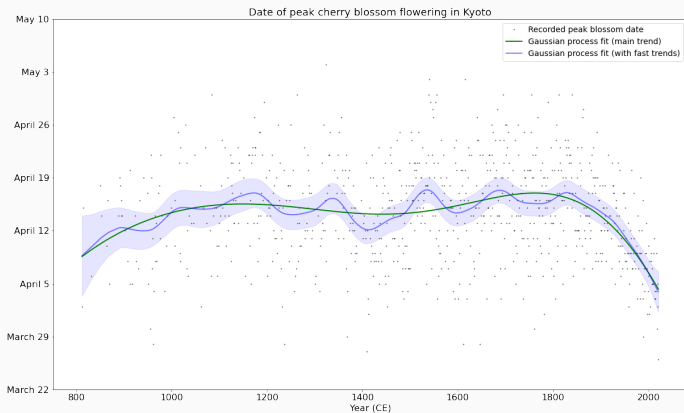

Cherry blossom regression

Plot only the slow trend:



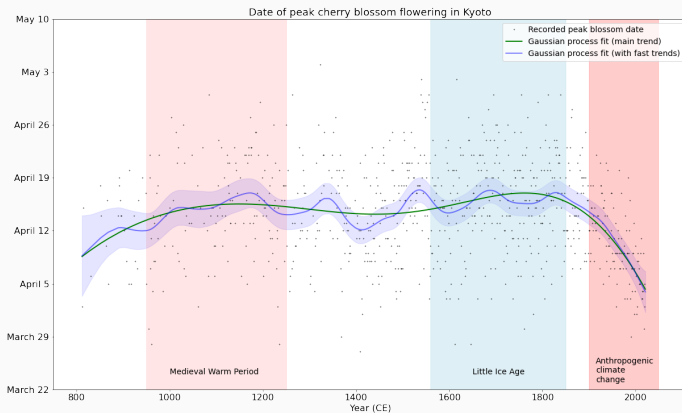
Cherry blossom regression

With the full fit:



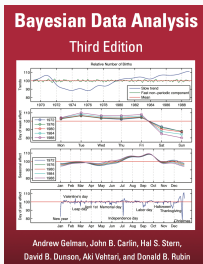
Cherry blossom regression

With the full fit:



Bigger example

For a big, complex example:



Ch. 21

PyMC3 does:

CO₂ at Mauna Loa
obs.

- Additive GP with 7 covariance kernels (and 2 non-GP components)
- Includes effects for long-term trends, rapid fluctuations, periodic effects at multiple scales

Summary

Summary:

- GPs allow flexible regression models without explicit function parameterization
- Covariance kernels determine the properties (smoothness, periodicity, etc.) of the functions in the prior
- Additive GPs allow decomposition of functions, modeling different scales of variation separately

Next week:

- Sequential state models
- Hidden Markov models