

Covariance and Gaussian processes

ISTA 410 / INFO 510: Bayesian Modeling and Inference

U. of Arizona School of Information

November 3, 2021

Last time:

- Bike share model
- Gaussian processes

Today:

- More details on Gaussian processes
- Late semester planning

GP: the definition

A Gaussian process is a random *function* – i.e., we're really talking about a probability distribution on a space of functions.

The feature that makes a GP a GP: if you pick any n values of x , then the vector of function values $(\mu(x_1), \mu(x_2), \dots, \mu(x_n))$ has a multivariate normal distribution:

$$(\mu(x_1), \dots, \mu(x_n)) \sim \text{Normal}((m(x_1), \dots, m(x_n)), K(x_1, \dots, x_n))$$

The GP is determined by its mean function m and covariance K .

GP: the definition

Typically, the covariance matrix is determined by a function called the *kernel* $k(x, x')$.

- $k(x, x')$ determines how much the value of $\mu(x)$ depends on $\mu(x')$.
- Common (not universal) property: $k(x, x')$ depends on the distance between x, x'
- Idea: we're looking for continuous functions, so the values of $\mu(x), \mu(x')$ should be close if x, x' are close; but if they're far apart they don't have to be

Squared exponential covariance

Very common choice: squared exponential covariance function:

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

Covariance is high when $x - x'$ is small, falls off at longer ranges.

Hyperparameters:

- η : the maximum covariance
- ℓ : the *length scale*, controls how quickly covariance decays

How this is realized in practice:

- We have a set of observations $f(x_i)$
- GP property says

$$f(x_i) \sim \text{MvNormal}((m(x_1), \dots, m(x_n)), K(x_1, \dots, x_n))$$

- So we evaluate the covariance function $k(x, x')$ at each pair of observed x values and use that to build a covariance matrix
- The Gaussian process distribution

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

is really a prior distribution on the space of continuous functions

Specifying a GP

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

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

- 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

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

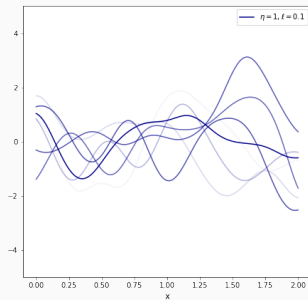
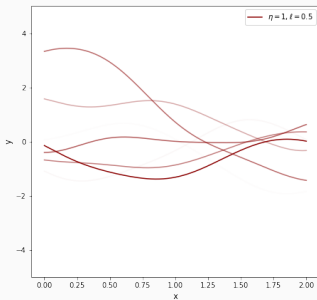
Choosing a covariance function

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



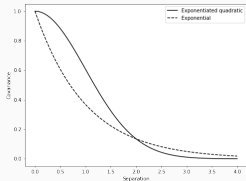
Last time: exponentiated quadratic vs exponential

- Exp. Quad:

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

- Exponential:

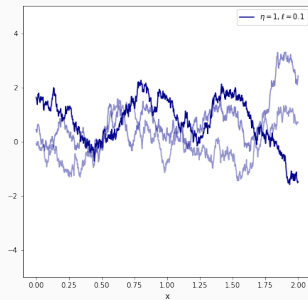
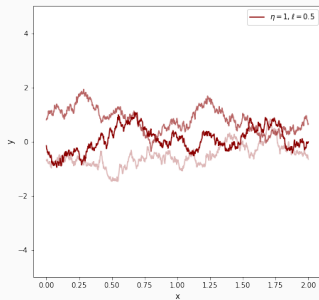
$$k(x, x') = \eta^2 \exp\left(-\frac{|x - x'|}{\ell}\right)$$



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

Smoothness

Samples from the GP prior: exponential covariance

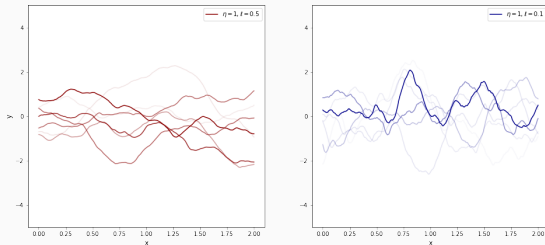


Smoothness interpolation:

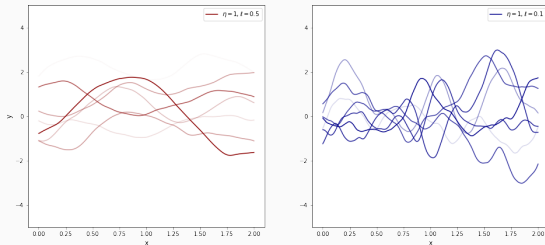
- Matern $_{\nu}$ kernel – $\nu = 1/2, 3/2, 5/2, \dots$
 - Formula built with gamma and Bessel functions
- $\nu = 1/2$ equivalent to exponential kernel
- $\nu \rightarrow \infty$ recovers exponentiated quadratic
- Draws from the prior have $\text{floor}(\nu)$ derivatives

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:

$$k(x, x') = \eta^2 \exp \left(-\frac{\sin^2(\pi|x - x'|/T)}{2\ell^2} \right)$$

- Linear kernel:

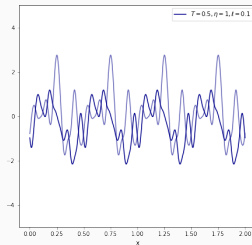
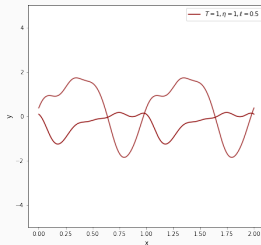
$$k(x, x') = (x - c)(x' - c)$$

- Polynomial kernel:

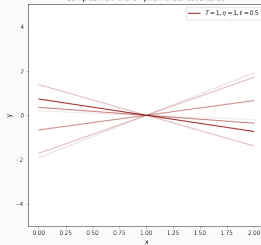
$$k(x, x') = [(x - c)(x' - c) + \textit{offset}]^d$$

Periodic

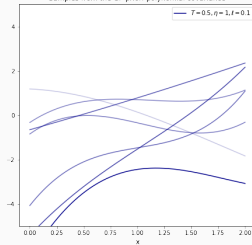
Samples from the GP prior: periodic covariance



Samples from the GP prior: linear covariance



Samples from the GP prior: polynomial covariance



PyMC3's gp submodule

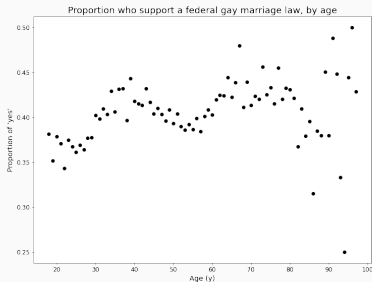
In PyMC3:

- Two implementations of Gaussian processes:
 - Marginal: a little simpler to use, but requires that the observed data is GP plus Gaussian noise
 - Latent: a little more flexible
- Many standard covariance functions:
 - Exponential, ExpQuad, Matern32, Matern52
 - Periodic, Linear

Example

Example from last time: data from the 2004 National Annenberg Election Survey

- % of respondents who favor a federal law legalizing gay marriage, as a function of age



Mathematical model

We can write the following model:

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

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

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

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

$$\ell \sim \text{Exponential}(0.2)$$

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

Notice similarity to a linear regression model!

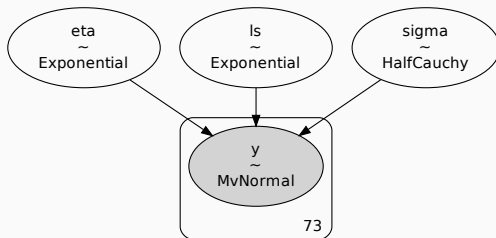
```
eta = pm.Exponential('eta', 1)
ls = pm.Exponential('ls', 0.2)
sigma = pm.HalfCauchy('sigma', 1)

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

y_ = gp.marginal_likelihood('y',
                             X=age, y=prop, noise=sigma)
# noise can be a covariance function too!
```

Example

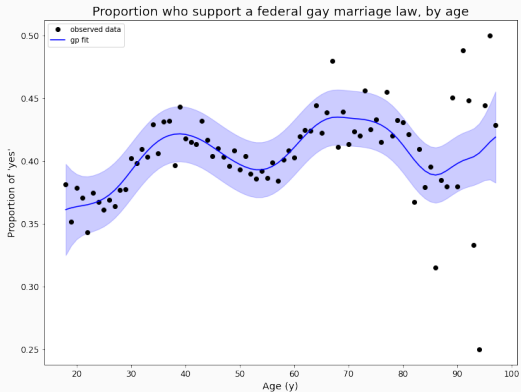
What does the plate diagram look like?



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

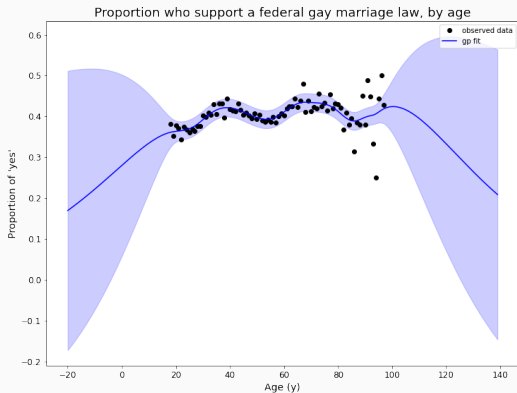
Example

Estimate MAP and use `gp.predict`



Example

What if we try to predict outside the range of the data?



What is happening here?

What happens inside `gp.predict`?

- We pass it a vector of new x values \tilde{x} and a vector of parameters (what we found with `find_MAP`)
- Conditional on the observed values, the posterior distribution of $\mu(\tilde{x})$ is multivariate normal
- `diag = True` specifies we only care about the marginal variance of each $\mu(\tilde{x})$
- Returns the mean and variance of the marginal distribution of each $\mu(\tilde{x})$

Rewriting the bicycle example

Rewriting the bike lane example has to work a bit differently:

- 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')
```

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')
```

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

Additive kernel example

Example: cherry blossom data

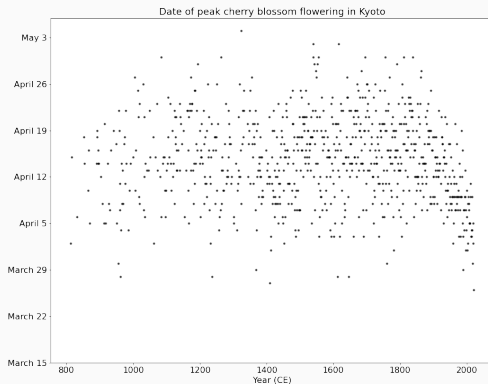
Cherry blossoms: big in Japan

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



Example: cherry blossom data

Plot the points:



Very noisy, but maybe there's a trend?

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)

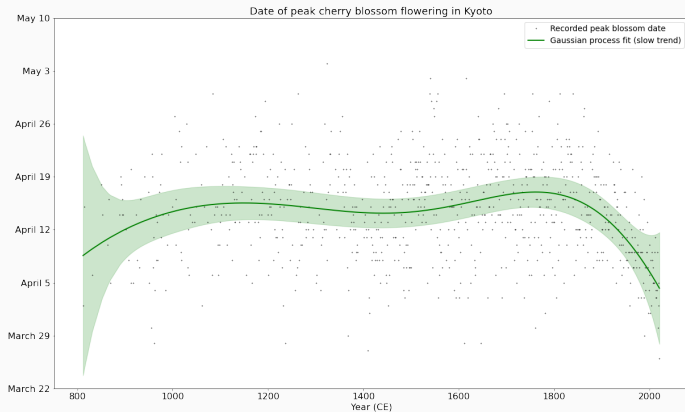
```
# Periodic component
eta_periodic = pm.Exponential('eta_periodic', 0.5)
l_periodic = pm.Gamma('l_periodic', 5, 1/25)
l_periodic_decay = pm.Gamma('l_periodic_decay', 5, 1/25)
period = pm.Exponential('period', 0.001)
periodic_cov = (eta_periodic ** 2)
                * pm.gp.cov.Periodic(1, ls=l_periodic,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

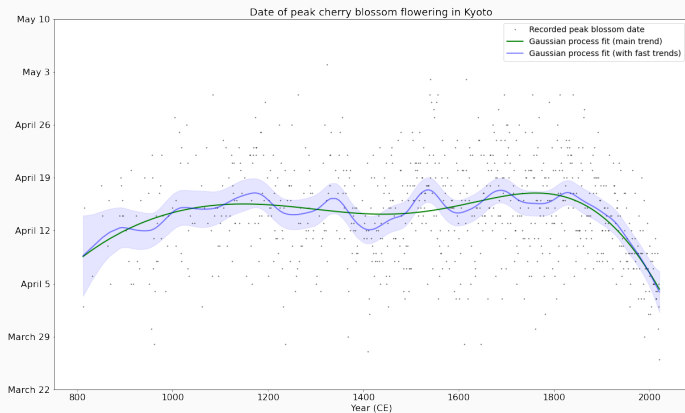
Cherry blossom regression

Plot only the slow trend:



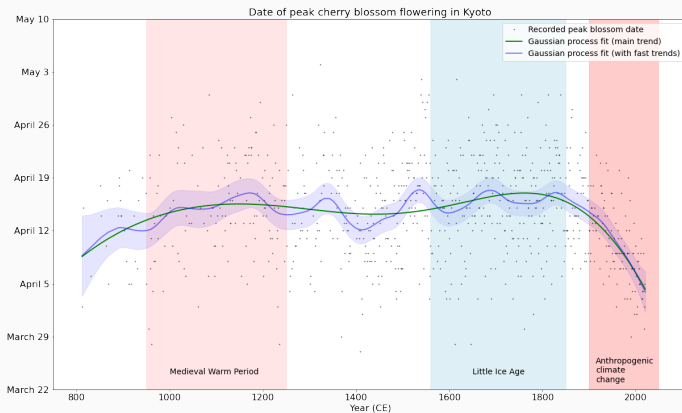
Cherry blossom regression

With the full fit:



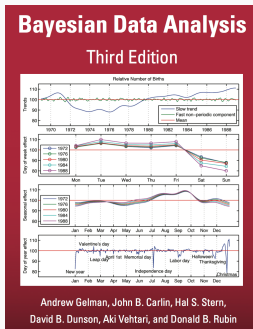
Cherry blossom regression

With the full fit:



Bigger example

For a big, complex example:



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

See BDA section 21.2 for details

More to be said on building covariance kernels for GPs:

- Can also multiply kernels
 - Think of multiplication as AND (vs addition as OR)
 - Product covariance is large only if both factors are large
- For more information / recommendations:
 - "The Kernel Cookbook"
 - <https://www.cs.toronto.edu/~duvenaud/cookbook/>

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

Timeline for the rest of the semester

Remaining time:

- 5 weeks, 10 classes (but really 9)
- Topics in the book:
 - Mixtures and “monsters”
 - Missing data
- Topics not in the book:
 - Hidden state models: Kalman filter, HMM
- Probably 4ish classes for extra topics

Additional topics

- Particle filters
- Bayesian neural networks
- Dirichlet processes
- Computational approaches other than MCMC (EM, expectation propagation, variational inference, etc.)
- ???

Summary

Today:

- Hierarchical linear regression models
 - Varying intercepts
 - Varying slopes

Next week:

- Multivariate normal mechanics
- Priors for covariance matrices
- Gaussian processes