Gaussian Processes for Regression: Models, Algorithms, and Applications, Day 2

Tamara Broderick
Associate Professor
MIT

http://www.tamarabroderick.com/tutorials.html

Roadmap

- Bayesian modeling and inference
- Gaussian process model
  - Popular version using a squared exponential kernel
- Gaussian process inference
  - Prediction & uncertainty quantification
- Observation noise
- What uncertainty are we quantifying?
- What can go wrong?
- Bayesian optimization

- Goals:
  - Learn the mechanism behind standard GPs to identify benefits and pitfalls (also in BayesOpt)
  - Learn the skills to be responsible users of standard GPs (transferable to other ML/AI methods)

Roadmap

- Bayesian modeling and inference
- Gaussian process model
  - Popular version using a squared exponential kernel
- Gaussian process inference
  - Prediction & uncertainty quantification
- Observation noise
- What uncertainty are we quantifying?
- What can go wrong?
- Bayesian optimization

- Goals:
  - Learn the mechanism behind standard GPs to identify benefits and pitfalls (also in BayesOpt)
  - Learn the skills to be responsible users of standard GPs (transferable to other ML/AI methods)

Roadmap

- Bayesian modeling and inference
- Gaussian process model
  - Popular version using a squared exponential kernel
- Gaussian process inference
  - Prediction & uncertainty quantification
- Observation noise
- What uncertainty are we quantifying?
- What can go wrong?
- Bayesian optimization

- Goals:
  - Learn the mechanism behind standard GPs to identify benefits and pitfalls (also in BayesOpt)
  - Learn the skills to be responsible users of standard GPs (transferable to other ML/AI methods)

Roadmap

- Bayesian modeling and inference
- Gaussian process model
  - Popular version using a squared exponential kernel
- Gaussian process inference
  - Prediction & uncertainty quantification
- Observation noise
- What uncertainty are we quantifying?
- What can go wrong?
- Bayesian optimization
- Goals:
  - Learn the mechanism behind standard GPs to identify benefits and pitfalls (also in BayesOpt)
  - Learn the skills to be responsible users of standard GPs (transferable to other ML/AI methods)

Roadmap

- Bayesian modeling and inference
- Gaussian process model
  - Popular version using a squared exponential kernel
- Gaussian process inference
  - Prediction & uncertainty quantification
- Observation noise
- What uncertainty are we quantifying?
- What can go wrong?
- Bayesian optimization

- Goals:
  - Learn the mechanism behind standard GPs to identify benefits and pitfalls (also in BayesOpt)
  - Learn the skills to be responsible users of standard GPs (transferable to other ML/AI methods)

Roadmap

- Bayesian modeling and inference
- Gaussian process model
  - Popular version using a squared exponential kernel
- Gaussian process inference
  - Prediction & uncertainty quantification
- Observation noise
- What uncertainty are we quantifying?
- What can go wrong?
- Bayesian optimization

- Goals:
  - Learn the mechanism behind standard GPs to identify benefits and pitfalls (also in BayesOpt)
  - Learn the skills to be responsible users of standard GPs (transferable to other ML/AI methods)

Roadmap

- Bayesian modeling and inference
- Gaussian process model
  - Popular version using a squared exponential kernel
- Gaussian process inference
  - Prediction & uncertainty quantification
- Observation noise
- What uncertainty are we quantifying?
- What can go wrong?
- Bayesian optimization

- Goals:
  - Learn the mechanism behind standard GPs to identify benefits and pitfalls (also in BayesOpt)
  - Learn the skills to be responsible users of standard GPs (transferable to other ML/AI methods)

Roadmap

- Bayesian modeling and inference
- Gaussian process model
  - Popular version using a squared exponential kernel
- Gaussian process inference
  - Prediction & uncertainty quantification
- Observation noise
- What uncertainty are we quantifying?
- What can go wrong?
- Bayesian optimization

- Goals:
  - Learn the mechanism behind standard GPs to identify benefits and pitfalls (also in BayesOpt)
  - Learn the skills to be responsible users of standard GPs (transferable to other ML/AI methods)

Roadmap

- Bayesian modeling and inference
- Gaussian process model
  - Popular version using a squared exponential kernel
- Gaussian process inference
  - Prediction & uncertainty quantification
- Observation noise
- What uncertainty are we quantifying?
- What can go wrong?
- Bayesian optimization

- Goals:
  - Learn the mechanism behind standard GPs to identify benefits and pitfalls (also in BayesOpt)
  - Learn the skills to be responsible users of standard GPs (transferable to other ML/AI methods)

Roadmap

- Bayesian modeling and inference
- Gaussian process model
  - Popular version using a squared exponential kernel
- Gaussian process inference
  - Prediction & uncertainty quantification
- Observation noise
- What uncertainty are we quantifying?
- What can go wrong?
- Bayesian optimization

- Goals:
  - Learn the mechanism behind standard GPs to identify benefits and pitfalls (also in BayesOpt)
  - Learn the skills to be responsible users of standard GPs (transferable to other ML/AI methods)

Roadmap

- Bayesian modeling and inference
- Gaussian process model
  - Popular version using a squared exponential kernel
- Gaussian process inference
  - Prediction & uncertainty quantification
- Observation noise
- What uncertainty are we quantifying?
- What can go wrong?
- Bayesian optimization

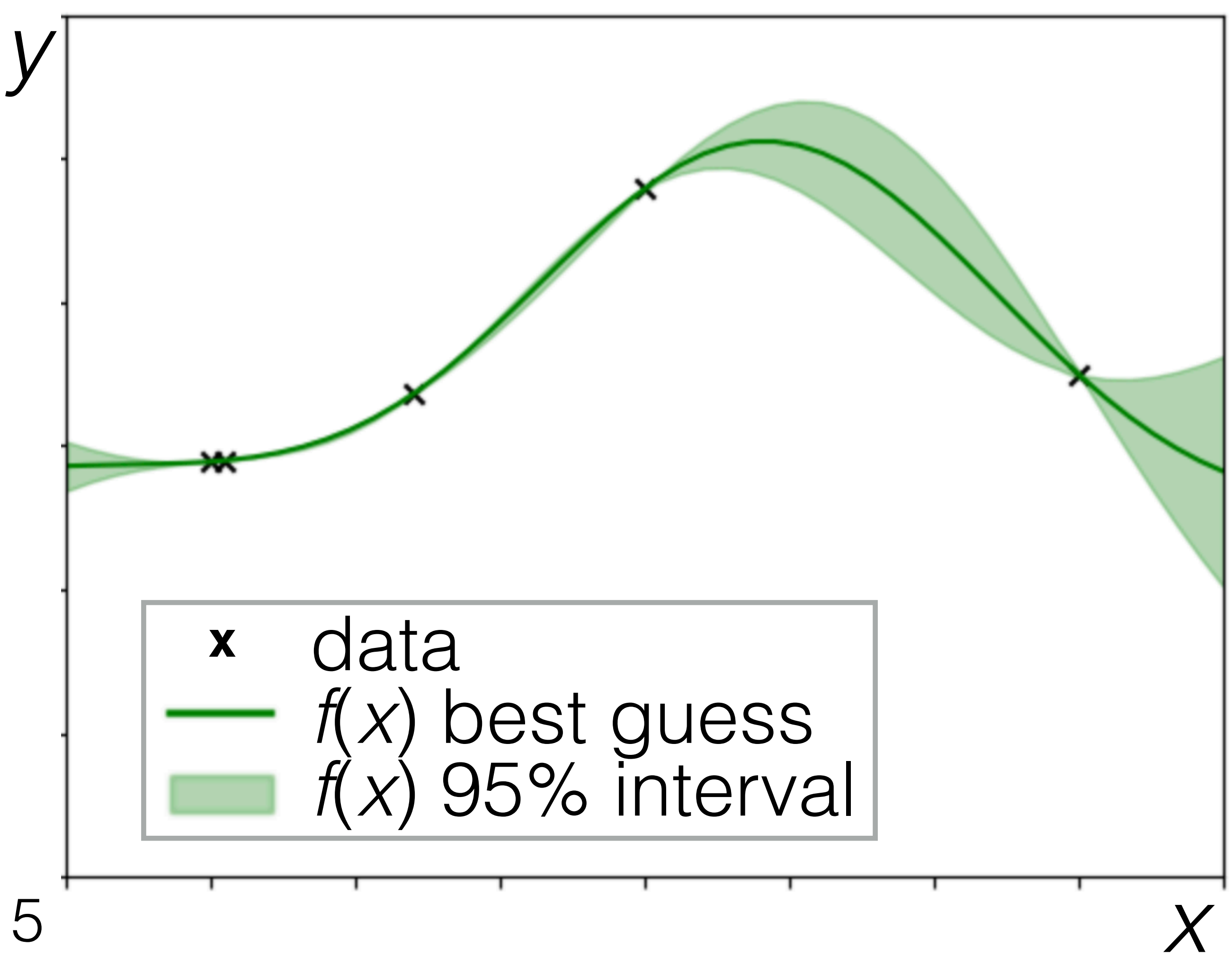
- Goals:
  - Learn the mechanism behind standard GPs to identify benefits and pitfalls (also in BayesOpt)
  - Learn the skills to be responsible users of standard GPs (transferable to other ML/AI methods)

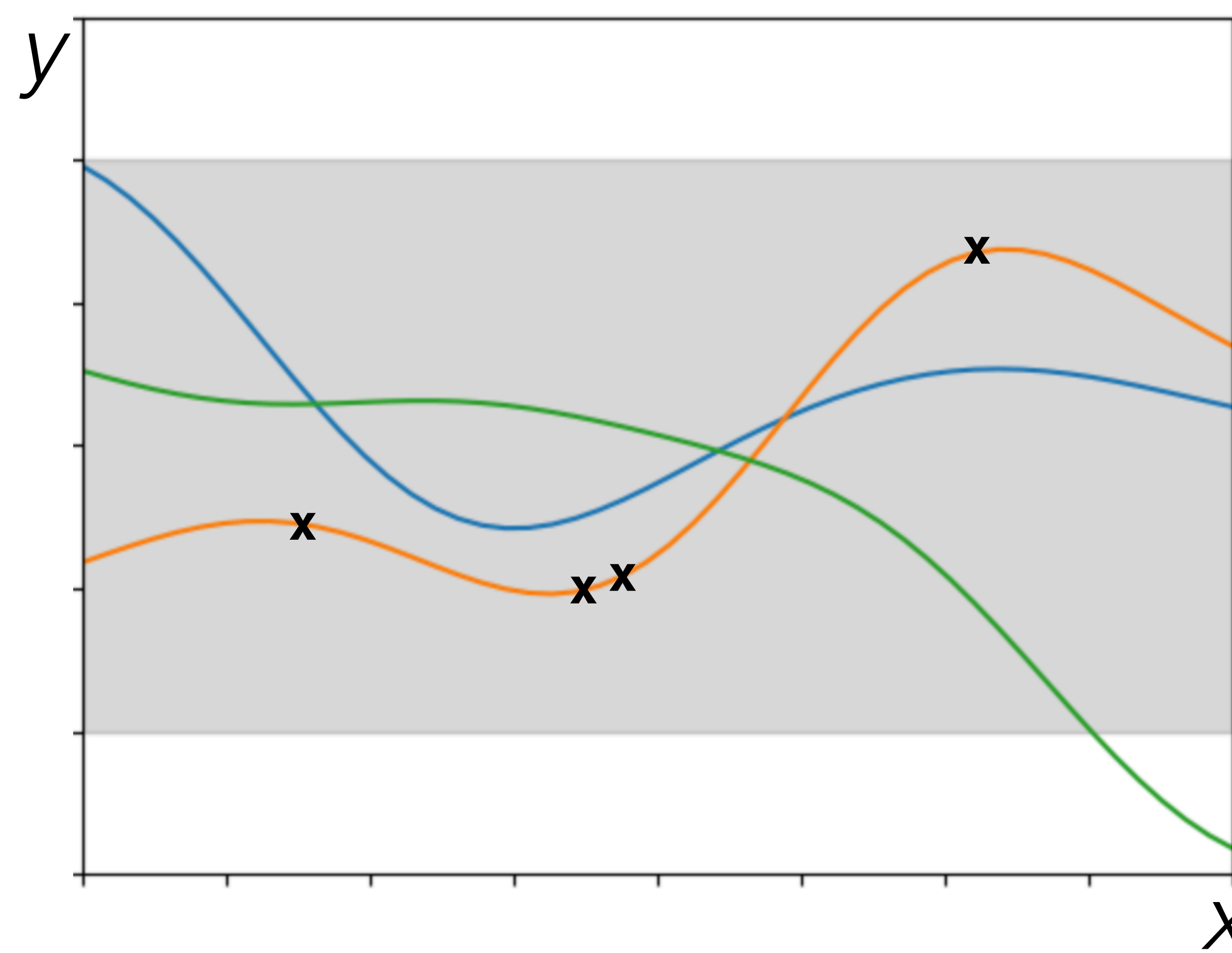
A Bayesian approach

- p(unknowns | data) $\propto$ p(data | unknowns) p(unknowns)

Given the data we've seen, what do we know about the underlying function?

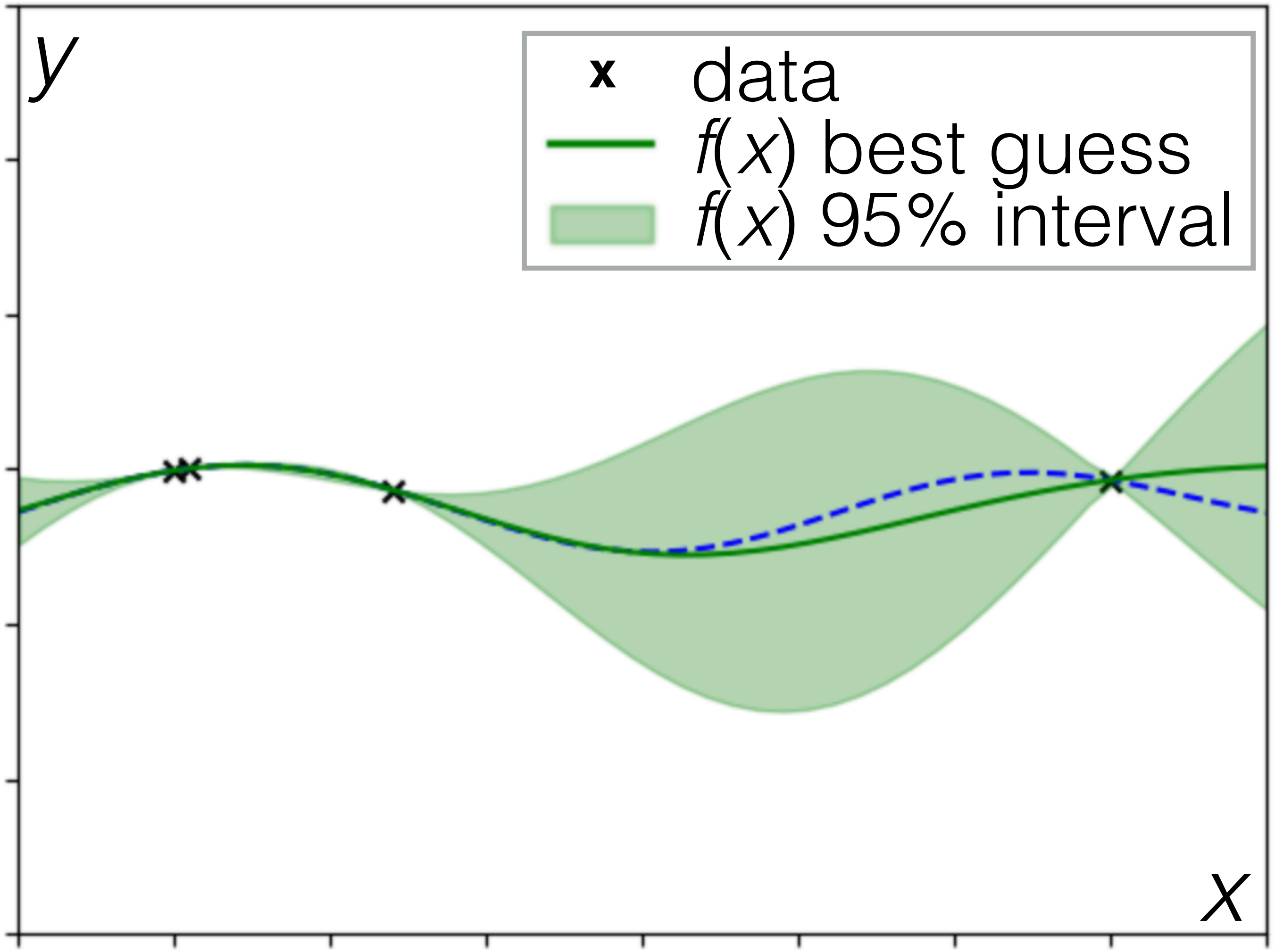
A (statistical) model that can generate functions and data of interest



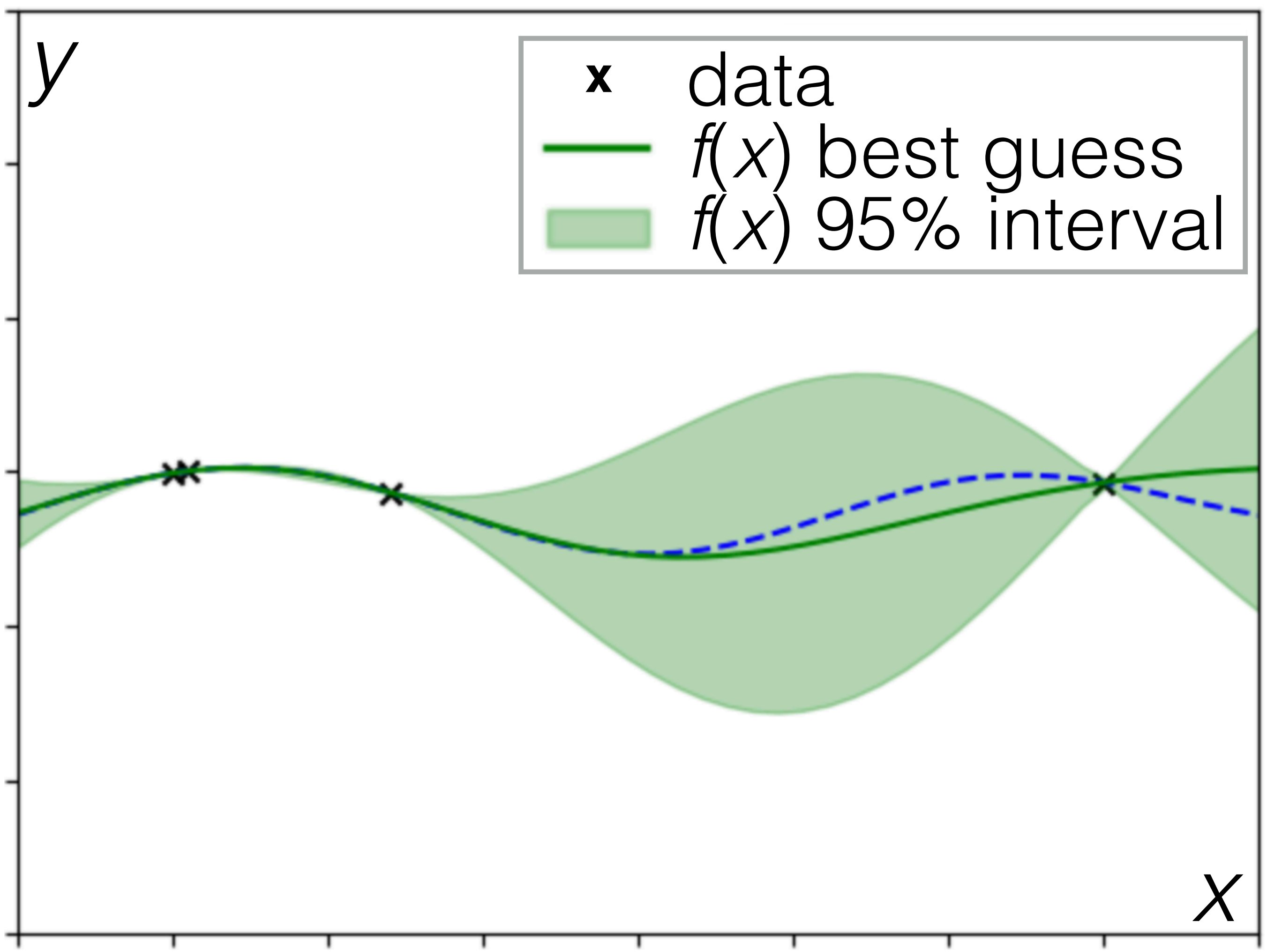


Closer look at the uncertainty interval

Closer look at the uncertainty interval

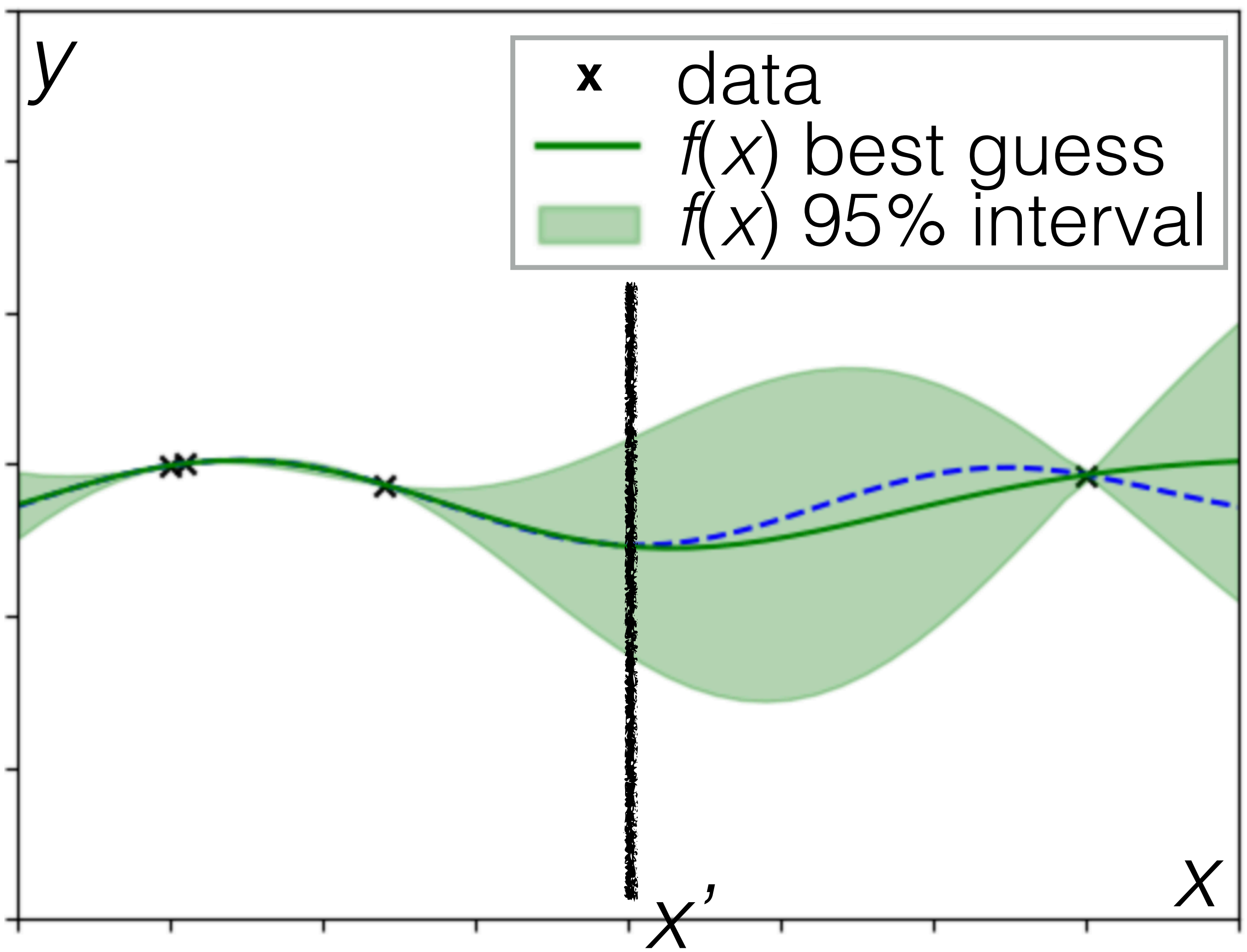


Closer look at the uncertainty interval



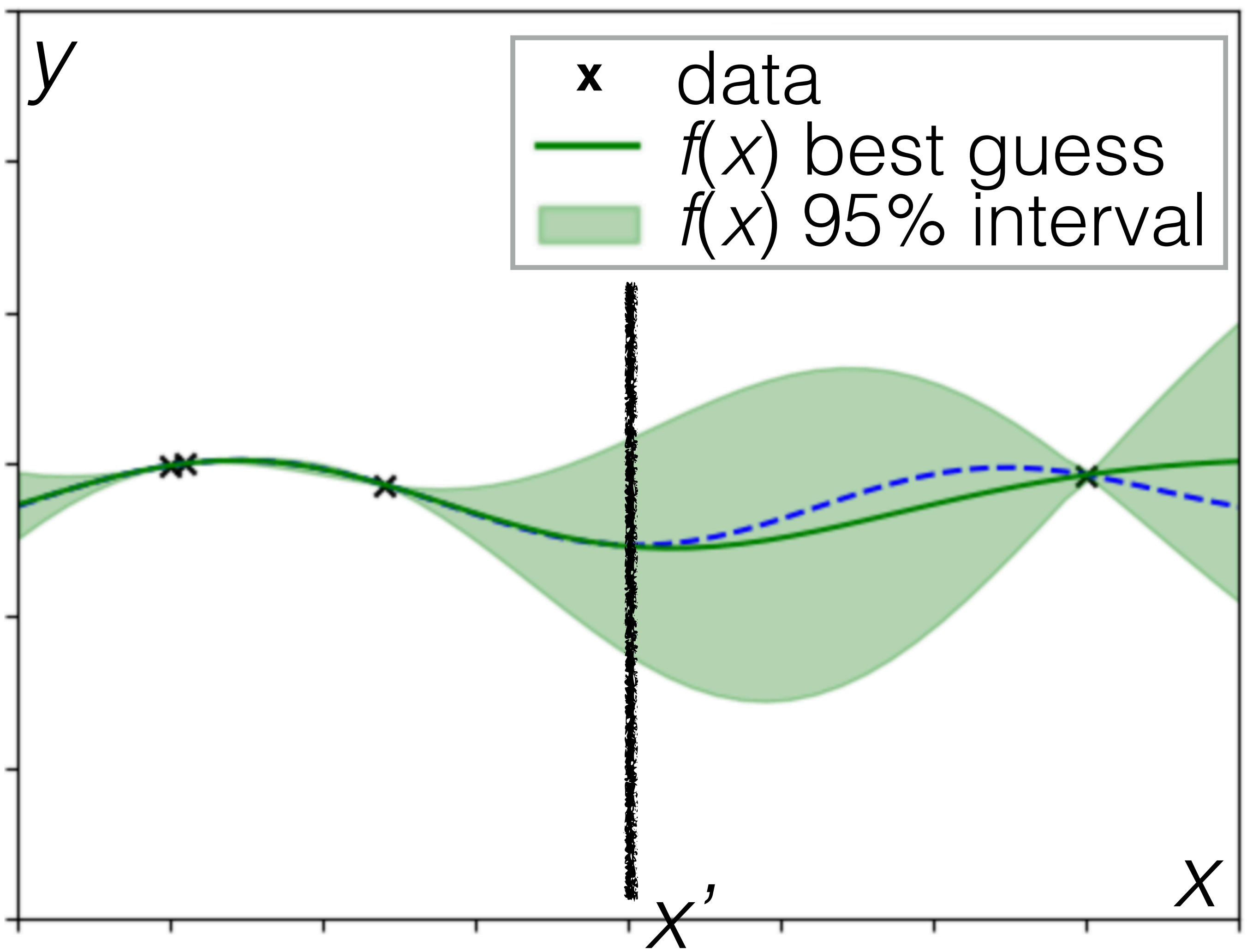
- Under GP, $f(x')|f(X), X, x'$ at a point *x'* is marginally Gaussian

Closer look at the uncertainty interval



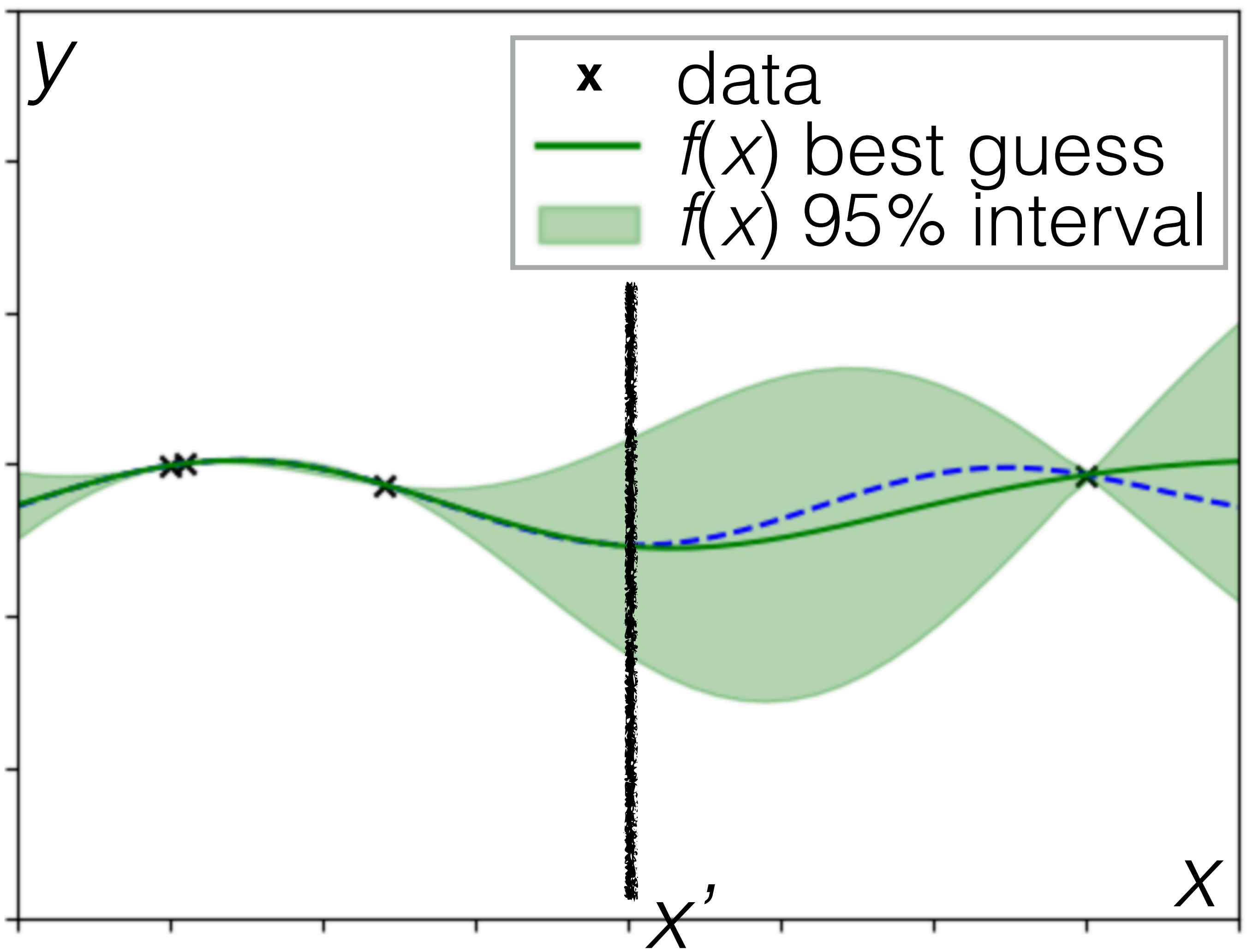
- Under GP, $f(x')|f(X), X, x'$ at a point *x'* is marginally Gaussian

Closer look at the uncertainty interval



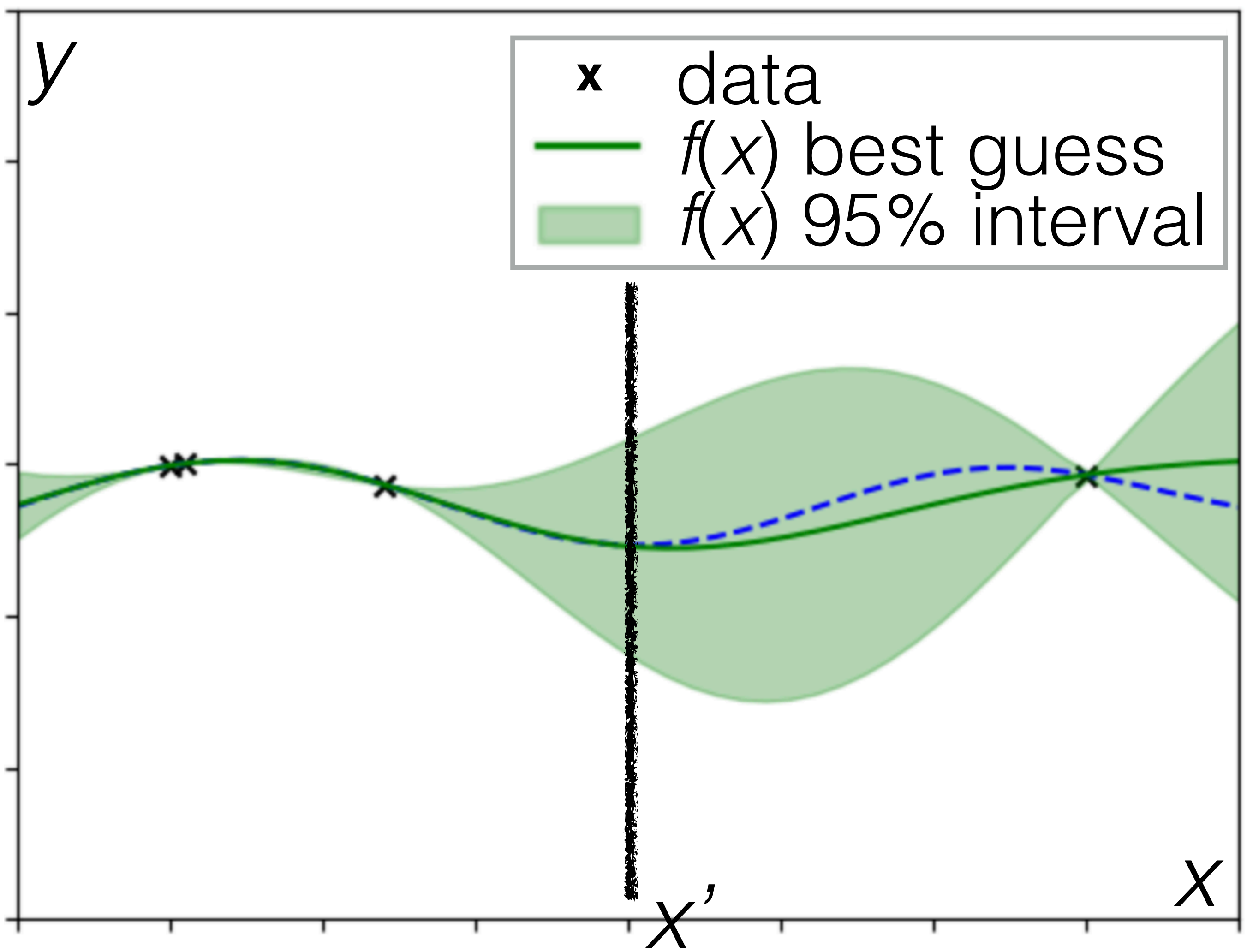
- Under GP, $f(x')|f(X), X, x'$ at a point *x'* is marginally Gaussian
- The green line at point *x'* is the mean of that Gaussian

Closer look at the uncertainty interval



- Under GP, $f(x')|f(X), X, x'$ at a point *x'* is marginally Gaussian
- The green line at point *x'* is the mean of that Gaussian
- The green interval at that point: mean +/- 2 std devs

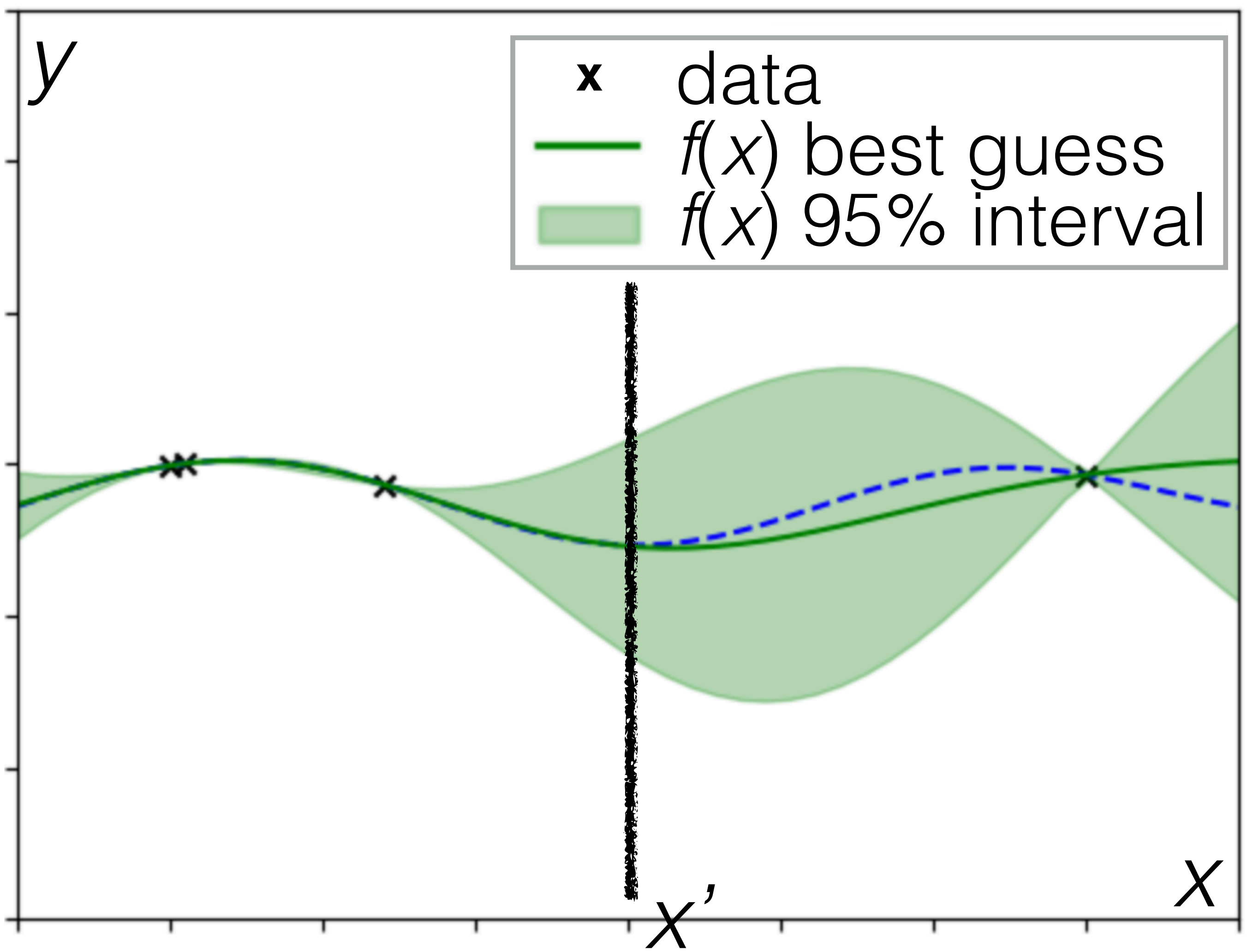
Closer look at the uncertainty interval



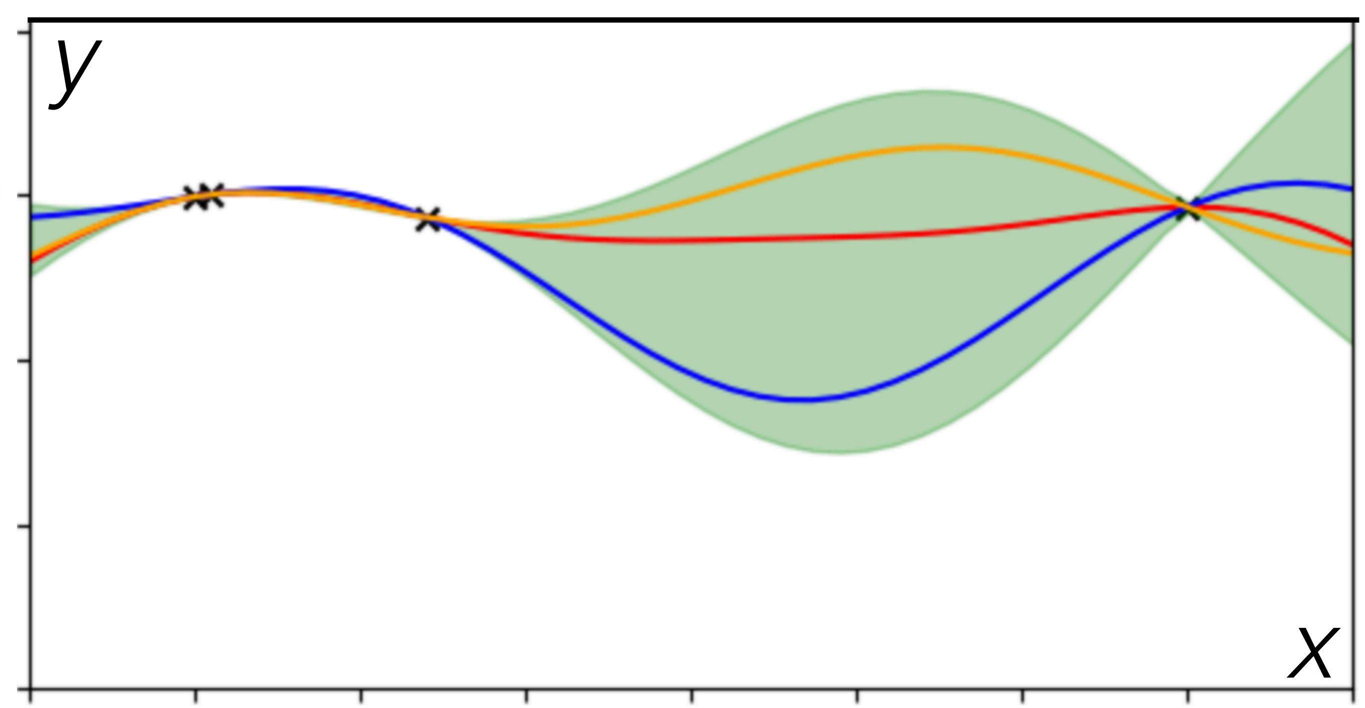
- Under GP, $f(x')|f(X), X, x'$ at a point *x'* is marginally Gaussian
- The green line at point *x'* is the mean of that Gaussian
- The green interval at that point: mean +/- 2 std devs

- Draw random *f* conditional on the training data

Closer look at the uncertainty interval

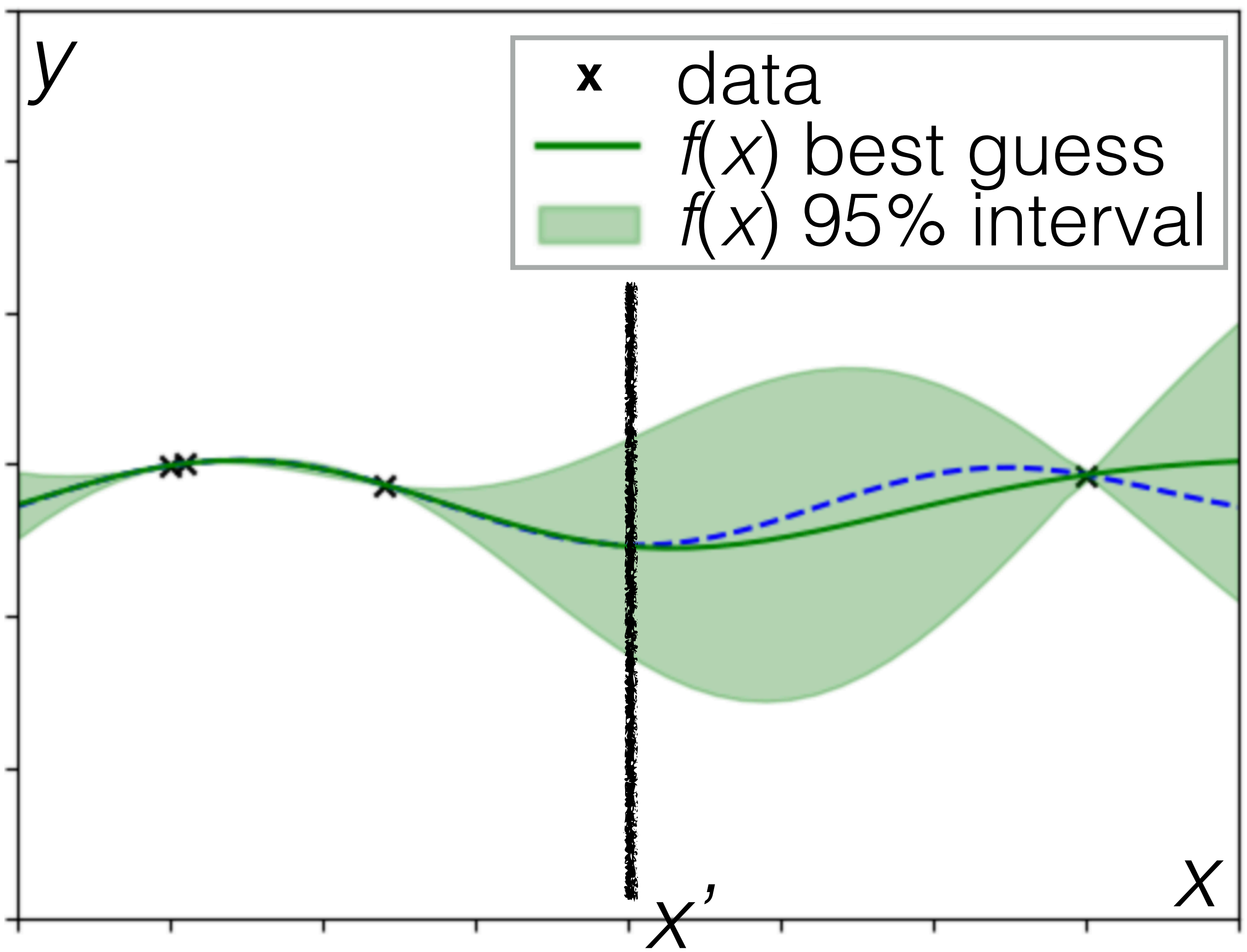


- Under GP, $f(x')|f(X), X, x'$ at a point *x'* is marginally Gaussian
- The green line at point *x'* is the mean of that Gaussian
- The green interval at that point: mean +/- 2 std devs

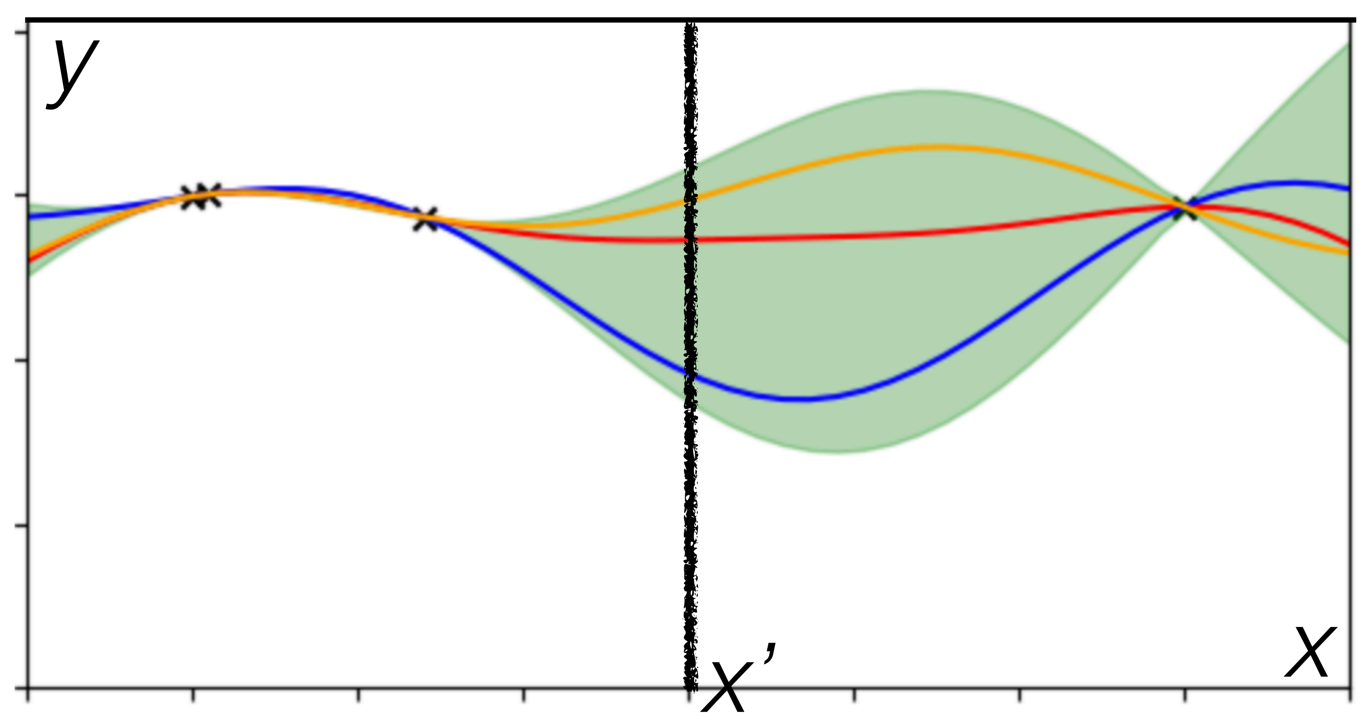


- Draw random *f* conditional on the training data

Closer look at the uncertainty interval

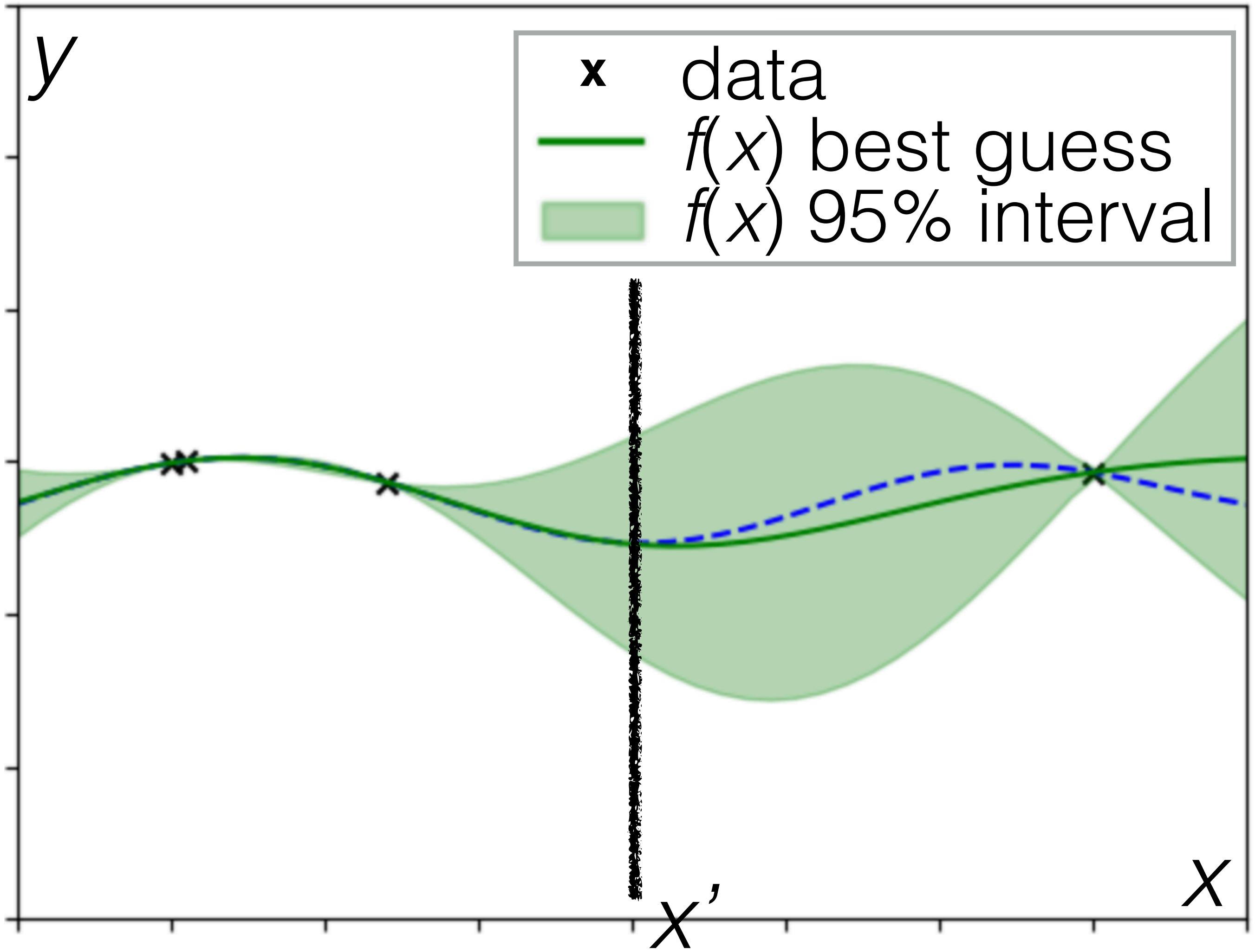


- Under GP, $f(x')|f(X), X, x'$ at a point *x'* is marginally Gaussian
- The green line at point *x'* is the mean of that Gaussian
- The green interval at that point: mean +/- 2 std devs

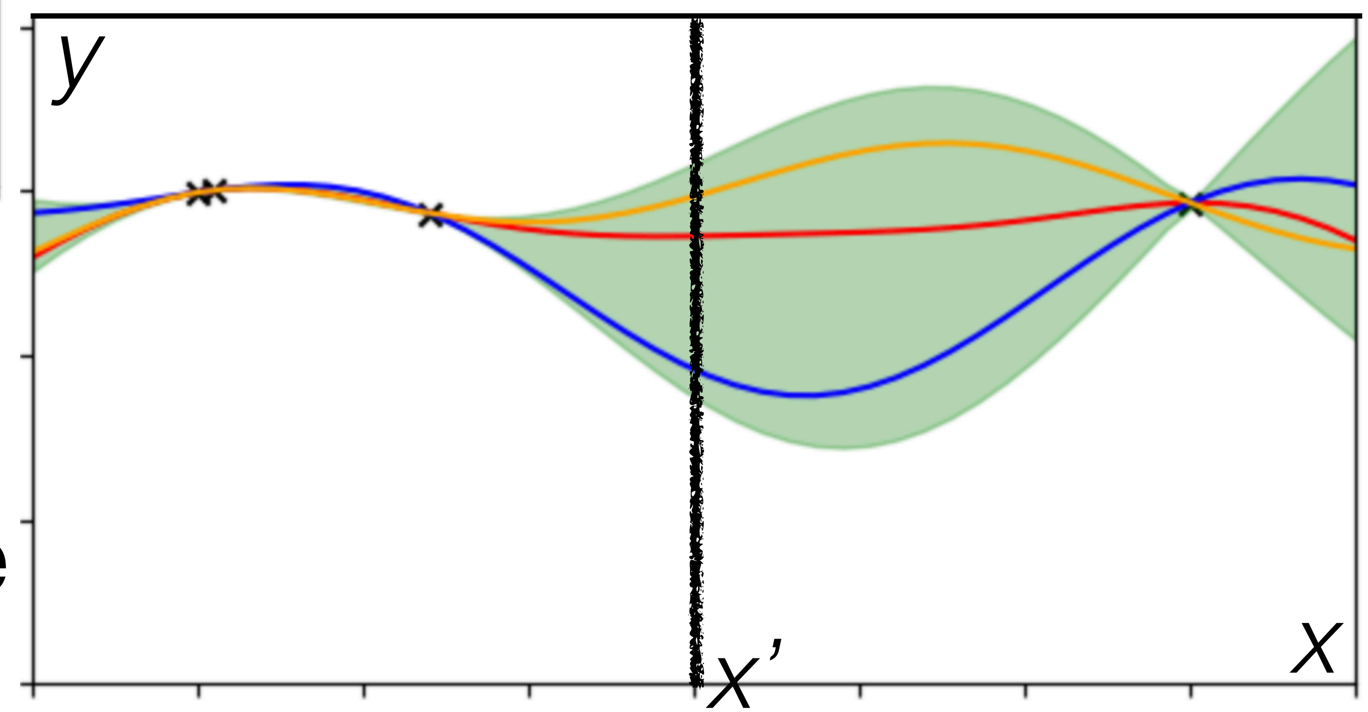


- Draw random *f* conditional on the training data

Closer look at the uncertainty interval

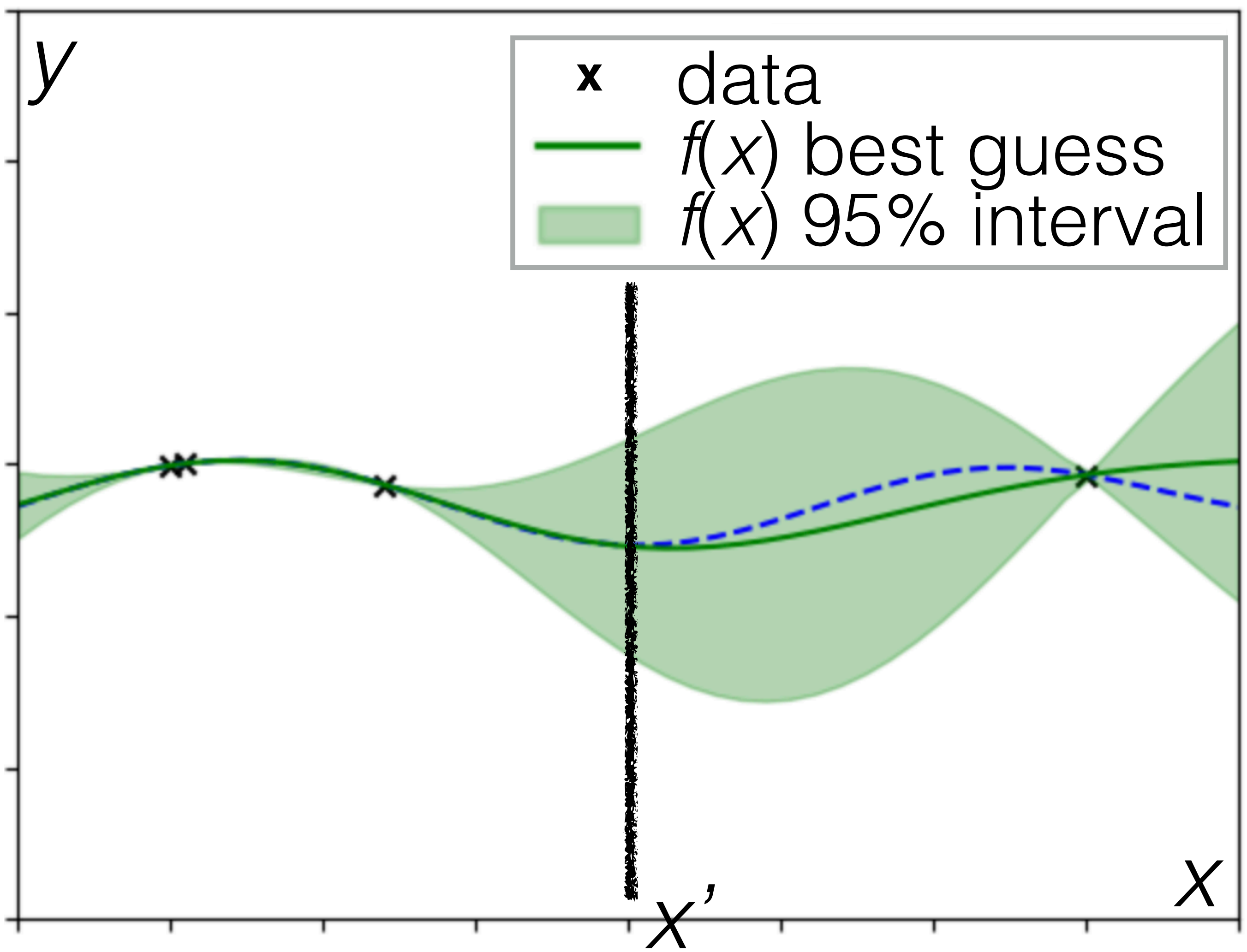


- Under GP, $f(x')|f(X), X, x'$ at a point *x'* is marginally Gaussian
- The green line at point *x'* is the mean of that Gaussian
- The green interval at that point: mean +/- 2 std devs

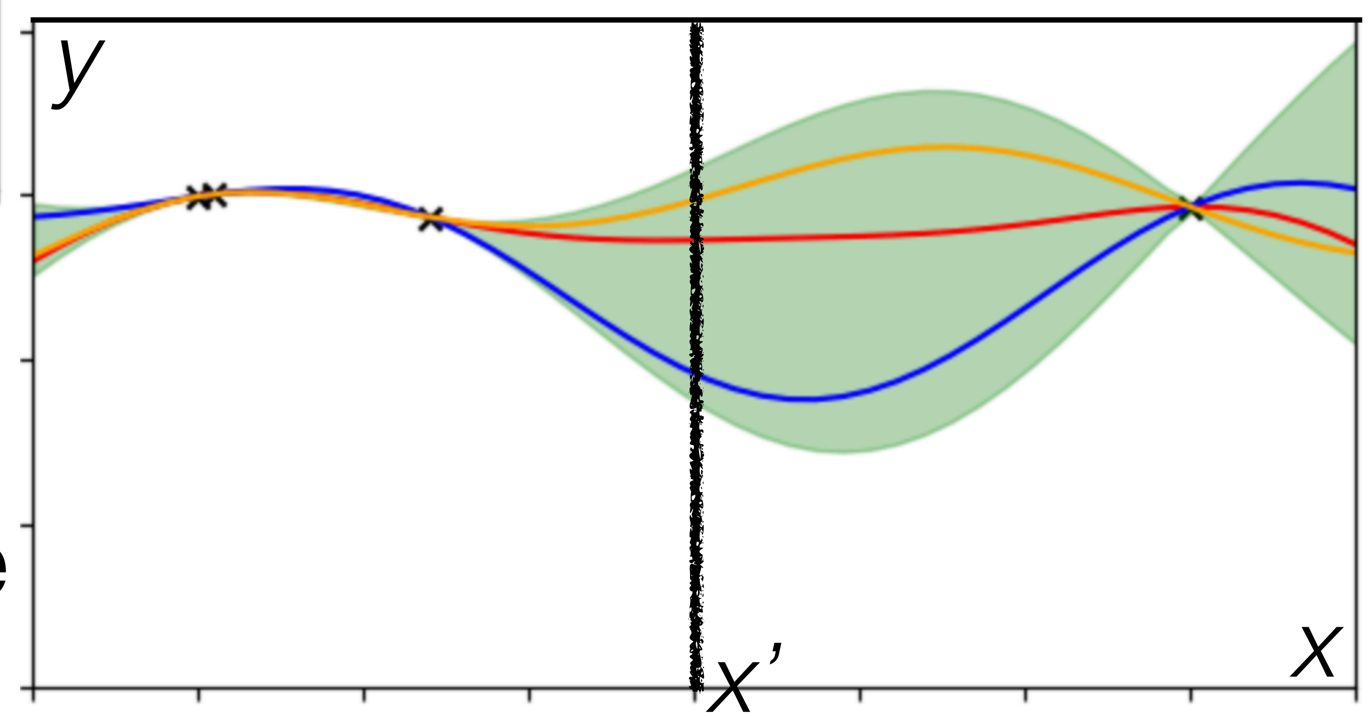


- Draw random *f* conditional on the training data
- Probability the draw is in the interval at *x'* is ?

Closer look at the uncertainty interval

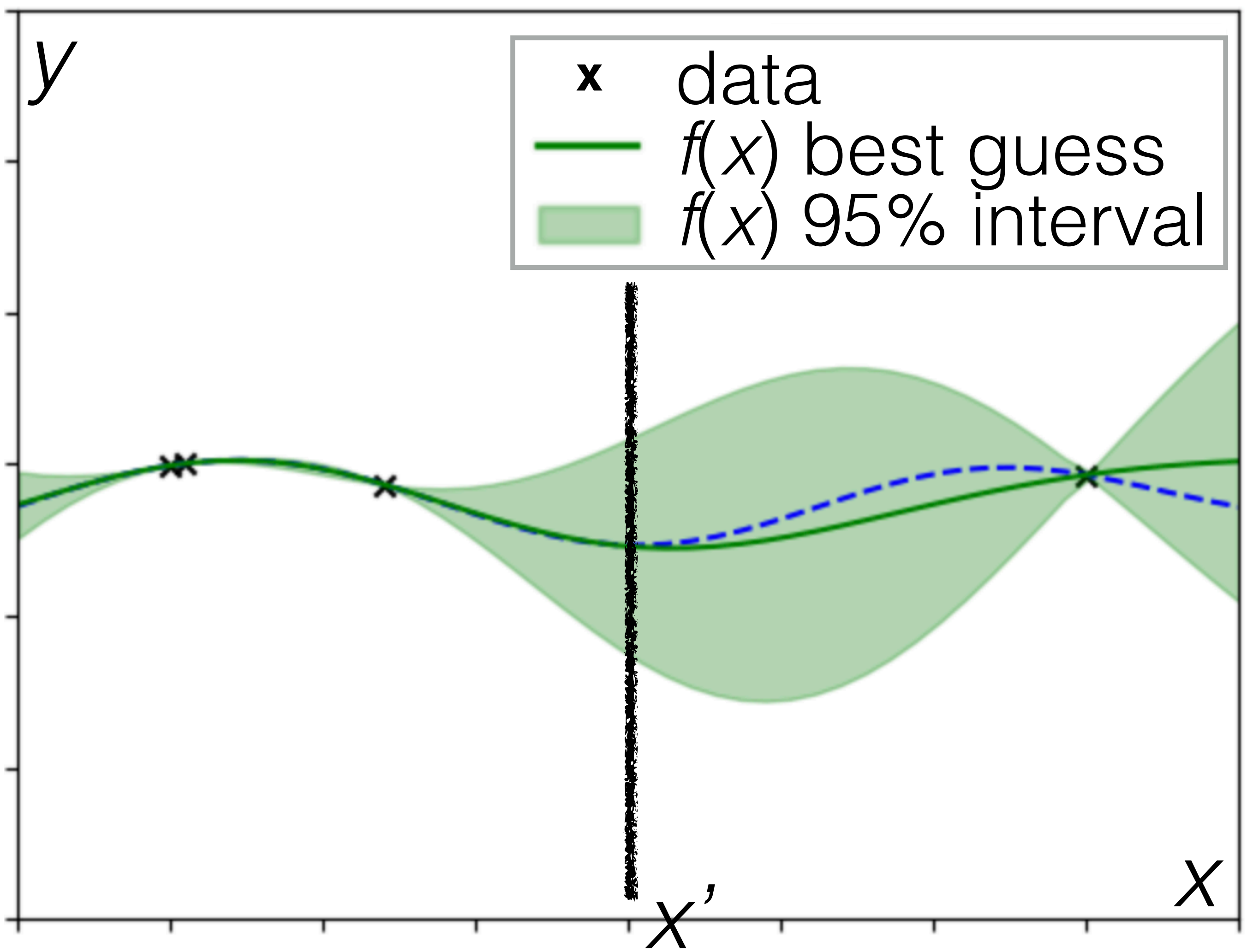


- Under GP, $f(x')|f(X), X, x'$ at a point *x'* is marginally Gaussian
- The green line at point *x'* is the mean of that Gaussian
- The green interval at that point: mean +/- 2 std devs

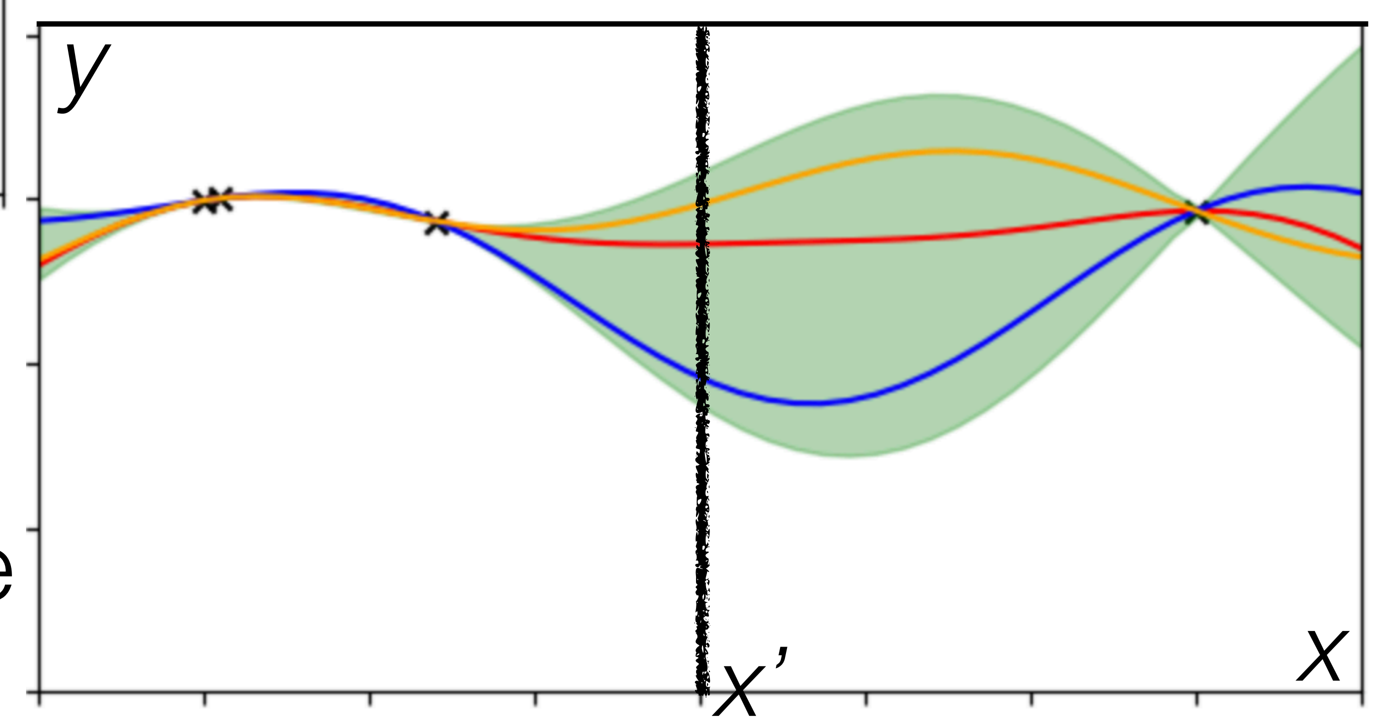


- Draw random *f* conditional on the training data
- Probability the draw is in the interval at *x'* is ~95%

Closer look at the uncertainty interval

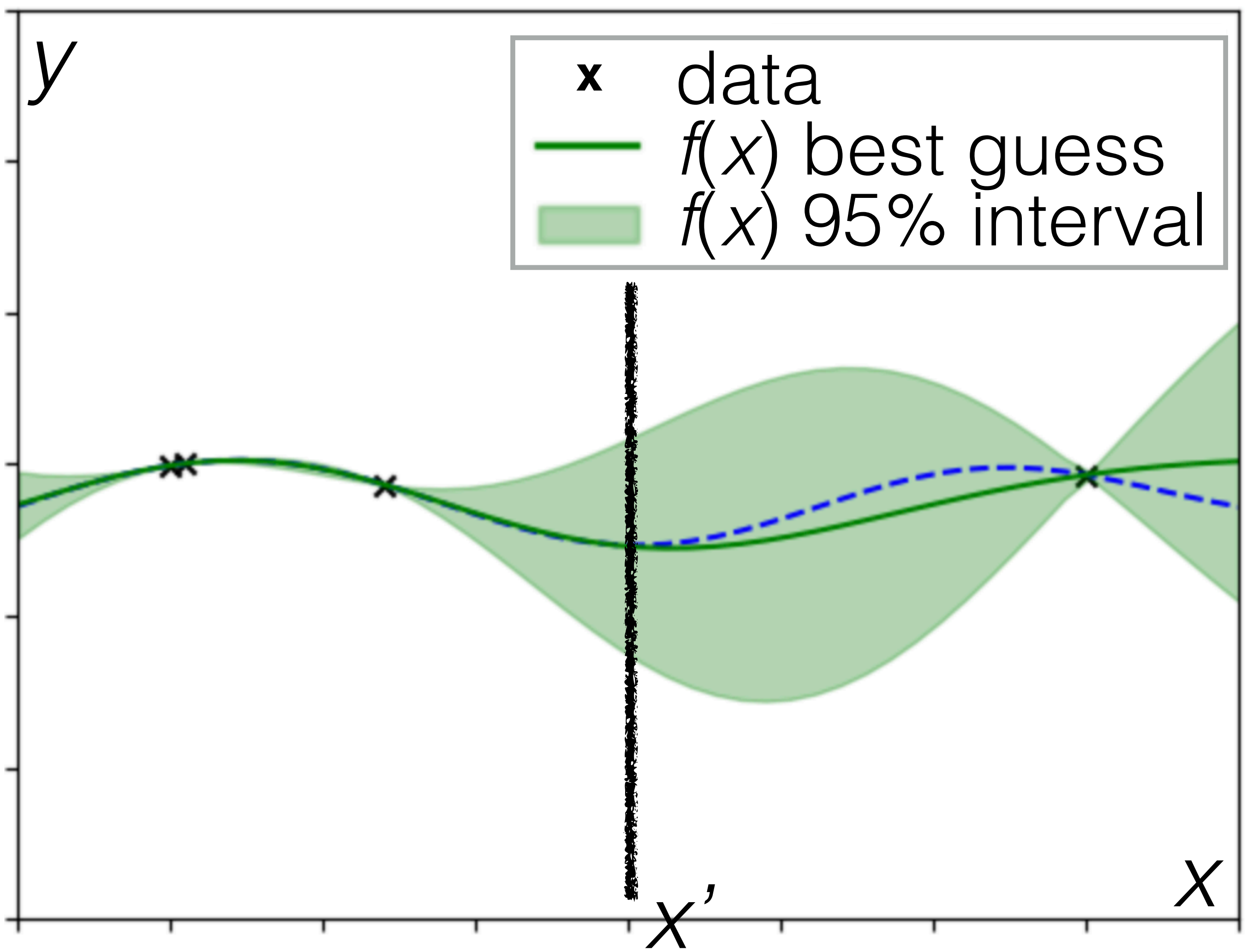


- Under GP, $f(x') | f(X), X, x'$ at a point *x'* is marginally Gaussian
- The green line at point *x'* is the mean of that Gaussian
- The green interval at that point: mean +/- 2 std devs

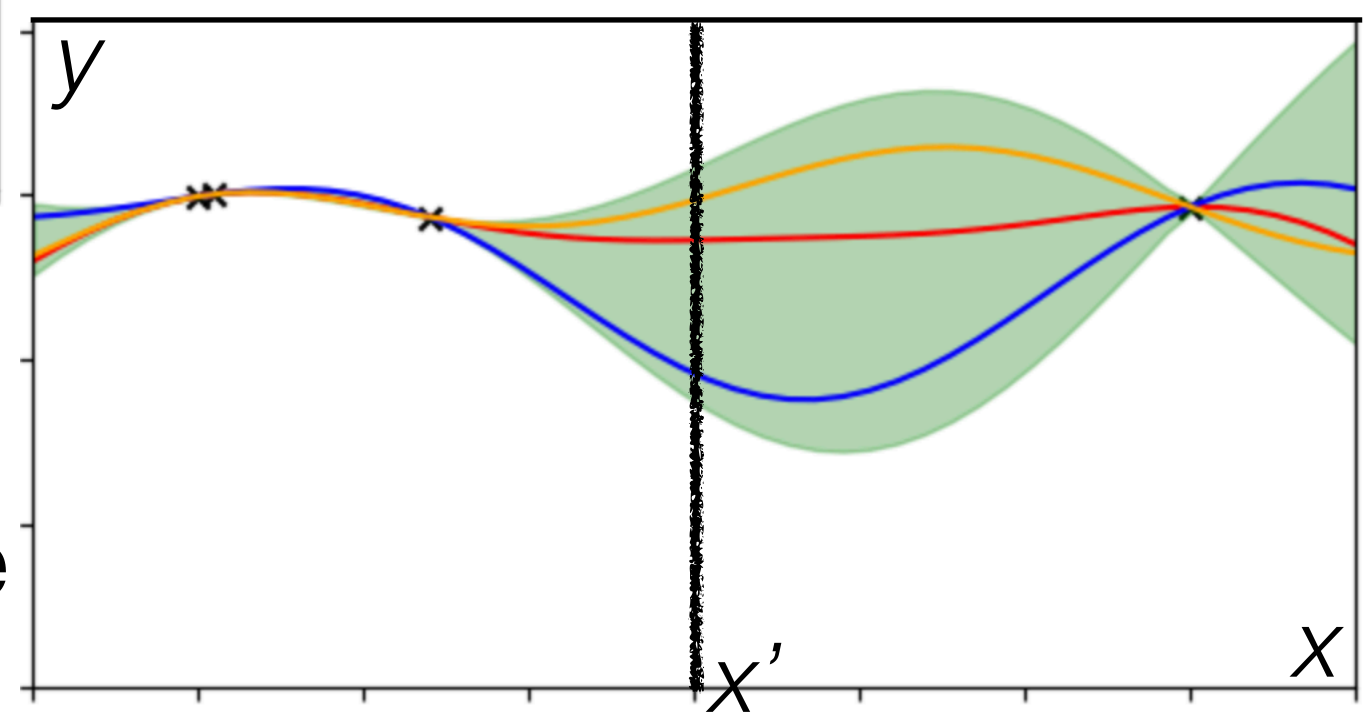


- Draw random *f* conditional on the training data
- Probability the draw is in the interval at *x'* is ~95%
- Probability that all points on *f* fall within the green interval across the whole plot ? ~95%

Closer look at the uncertainty interval

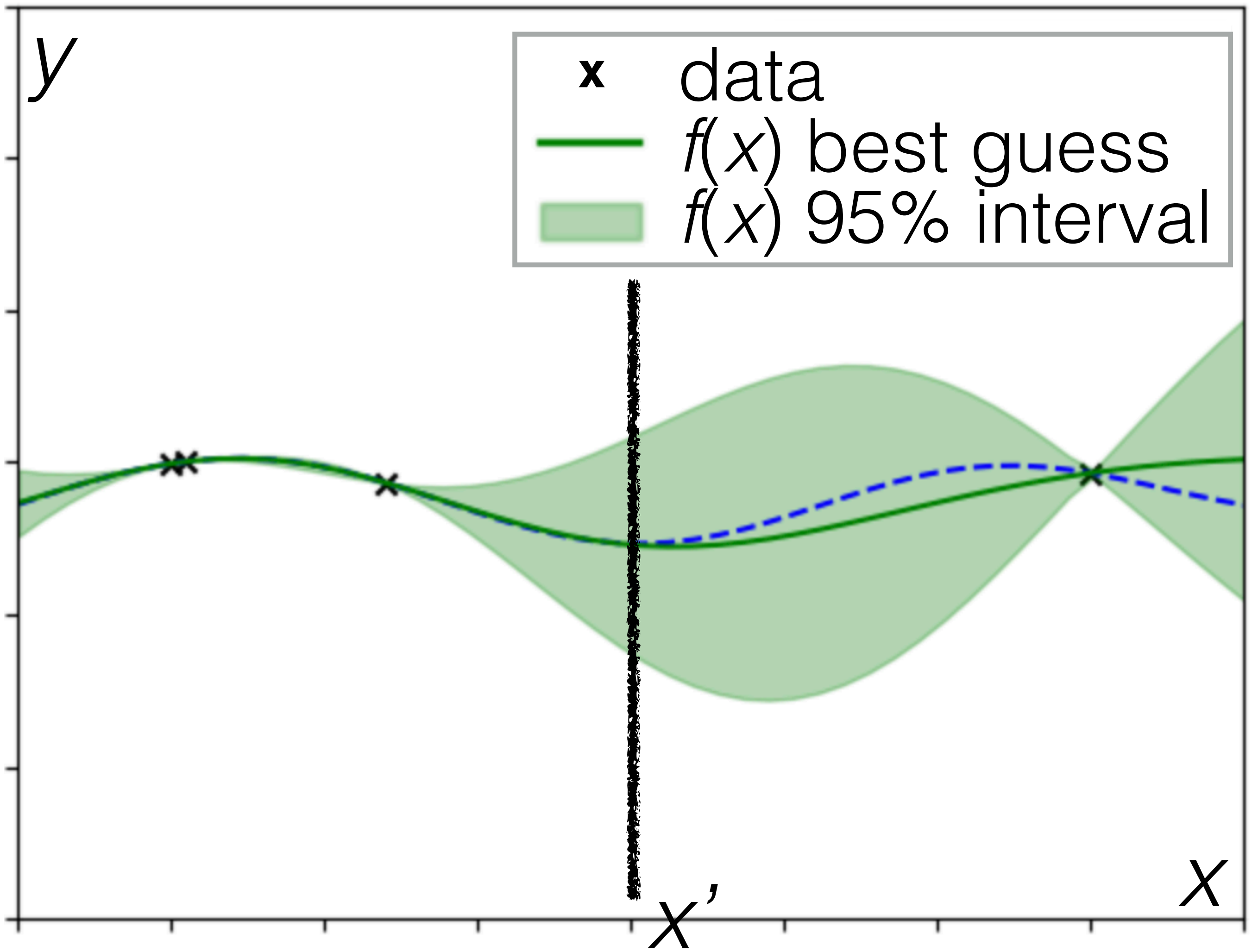


- Under GP, $f(x')|f(X), X, x'$ at a point *x'* is marginally Gaussian
- The green line at point *x'* is the mean of that Gaussian
- The green interval at that point: mean +/- 2 std devs

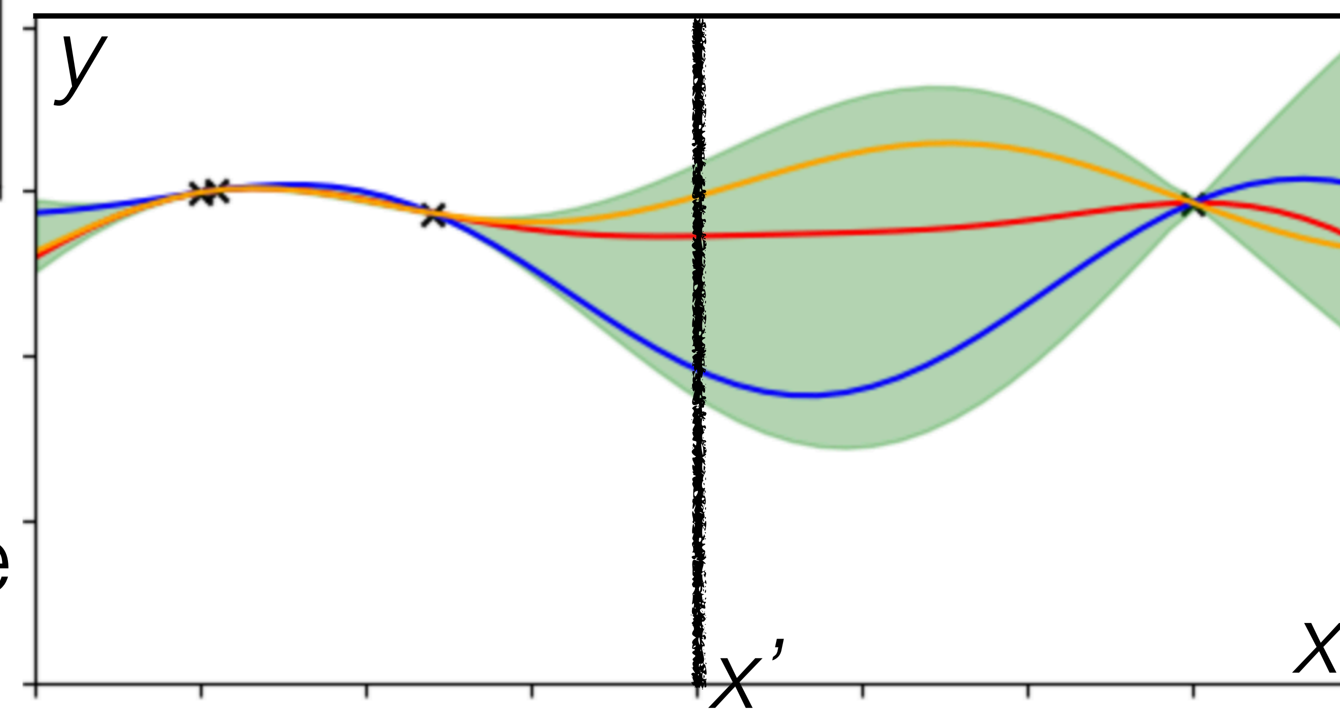


- Draw random *f* conditional on the training data
- Probability the draw is in the interval at *x'* is ~95%
- Probability that all points on *f* fall within the green interval across the whole plot will generally *not* be ~95%

Closer look at the uncertainty interval



- Under GP, $f(x')|f(X), X, x'$ at a point *x'* is marginally Gaussian
- The green line at point *x'* is the mean of that Gaussian
- The green interval at that point: mean +/- 2 std devs



- Draw random *f* conditional on the training data
- Probability the draw is in the interval at *x'* is ~95%
- Probability that all points on *f* fall within the green interval across the whole plot will generally *not* be ~95%

Squared exponential kernel revisited

Squared exponential kernel revisited

- What if we happened to measure our data on a different scale? [demo]

Squared exponential kernel revisited

- What if we happened to measure our data on a different scale? [demo]
- We've been using this particular kernel:

$$k(x, x') = \sigma^2 \exp(-\tfrac{1}{2}(x - x')^2), \sigma = 1$$

Squared exponential kernel revisited

- What if we happened to measure our data on a different scale? [demo]
- We've been using this particular kernel:

$$k(x, x') = \sigma^2 \exp(-\tfrac{1}{2}(x - x')^2), \sigma = 1$$

  - What do we expect from the scale of $f(x)$ a priori?

Squared exponential kernel revisited

- What if we happened to measure our data on a different scale? [demo]
- We've been using this particular kernel:

$$k(x, x') = \sigma^2 \exp(-\tfrac{1}{2}(x - x')^2), \sigma = 1$$

  - What do we expect from the scale of *f*(*x*) a priori?
    - At one *x*, with ~95% probability a priori, $f(x) \in$?

Squared exponential kernel revisited

- What if we happened to measure our data on a different scale? [demo]
- We've been using this particular kernel:

$$k(x, x') = \sigma^2 \exp(-\tfrac{1}{2}(x - x')^2), \sigma = 1$$

  - What do we expect from the scale of *f*(*x*) a priori?
    - At one *x*, with ~95% probability a priori, $f(x) \in (-2, 2)$

Squared exponential kernel revisited

- What if we happened to measure our data on a different scale? [demo]
- We've been using this particular kernel:

$$k(x, x') = \sigma^2 \exp(-\tfrac{1}{2}(x - x')^2), \sigma = 1$$

  - What do we expect from the scale of *f*(*x*) a priori?
    - At one *x*, with ~95% probability a priori, $f(x) \in (-2, 2)$
    - Marginal variance cannot increase with data

Squared exponential kernel revisited

- What if we happened to measure our data on a different scale? [demo1, demo2]
- We've been using this particular kernel:

$$k(x, x') = \sigma^2 \exp(-\tfrac{1}{2}(x - x')^2), \sigma = 1$$

  - What do we expect from the scale of *f*(*x*) a priori?
    - At one *x*, with ~95% probability a priori, $f(x) \in (-2, 2)$
    - Marginal variance cannot increase with data

Squared exponential kernel revisited

- What if we happened to measure our data on a different scale? [demo1, demo2]
- We've been using this particular kernel:

$$k(x, x') = \sigma^2 \exp(-\tfrac{1}{2}(x - x')^2), \sigma = 1$$

  - What do we expect from the scale of *f*(*x*) a priori?
    - At one *x*, with ~95% probability a priori, $f(x) \in (-2, 2)$
    - Marginal variance cannot increase with data
  - What counts as "close" in *x*?

Squared exponential kernel revisited

- What if we happened to measure our data on a different scale? [demo1, demo2]
- We've been using this particular kernel:

$$k(x, x') = \sigma^2 \exp(-\tfrac{1}{2}(x - x')^2), \sigma = 1$$

  - What do we expect from the scale of *f*(*x*) a priori?
    - At one *x*, with ~95% probability a priori, $f(x) \in (-2, 2)$
    - Marginal variance cannot increase with data
  - What counts as "close" in *x*?

Squared exponential kernel revisited

- What if we happened to measure our data on a different scale? [demo1, demo2]
- We've been using this particular kernel:

$$k(x, x') = \sigma^2 \exp(-\tfrac{1}{2}(x - x')^2), \sigma = 1$$

  - What do we expect from the scale of *f*(*x*) a priori?
    - At one *x*, with ~95% probability a priori, $f(x) \in (-2, 2)$
    - Marginal variance cannot increase with data
  - What counts as "close" in *x*?

$$\exp(-\tfrac{1}{2}2^2) \approx 0.14 \quad \exp(-\tfrac{1}{2}3^2) \approx 0.011 \quad \exp(-\tfrac{1}{2}4^2) \approx 0.00034$$

Squared exponential kernel revisited

- What if we happened to measure our data on a different scale? [demo1, demo2]
- We've been using this particular kernel:

$$k(x, x') = \sigma^2 \exp(-\tfrac{1}{2}(x - x')^2), \sigma = 1$$

  - What do we expect from the scale of *f*(*x*) a priori?
    - At one *x*, with ~95% probability a priori, $f(x) \in (-2, 2)$
    - Marginal variance cannot increase with data
  - What counts as "close" in *x*?

$$\exp(-\tfrac{1}{2}2^2) \approx 0.14 \quad \exp(-\tfrac{1}{2}3^2) \approx 0.011 \quad \exp(-\tfrac{1}{2}4^2) \approx 0.00034$$

- What can we do to handle different *x* and *f*(*x*) scales?

Squared exponential kernel revisited

- What if we happened to measure our data on a different scale? [demo1, demo2]
- We've been using this particular kernel:

$$k(x, x') = \sigma^2 \exp(-\tfrac{1}{2}(x - x')^2), \sigma = 1$$

  - What do we expect from the scale of *f*(*x*) a priori?
    - At one *x*, with ~95% probability a priori, $f(x) \in (-2, 2)$
    - Marginal variance cannot increase with data
  - What counts as "close" in *x*?

$$\exp(-\tfrac{1}{2}2^2) \approx 0.14 \quad \exp(-\tfrac{1}{2}3^2) \approx 0.011 \quad \exp(-\tfrac{1}{2}4^2) \approx 0.00034$$

- What can we do to handle different *x* and *f*(*x*) scales?
  - Normalization in *y* can help; in *x*, can still be hiccups

Squared exponential kernel revisited

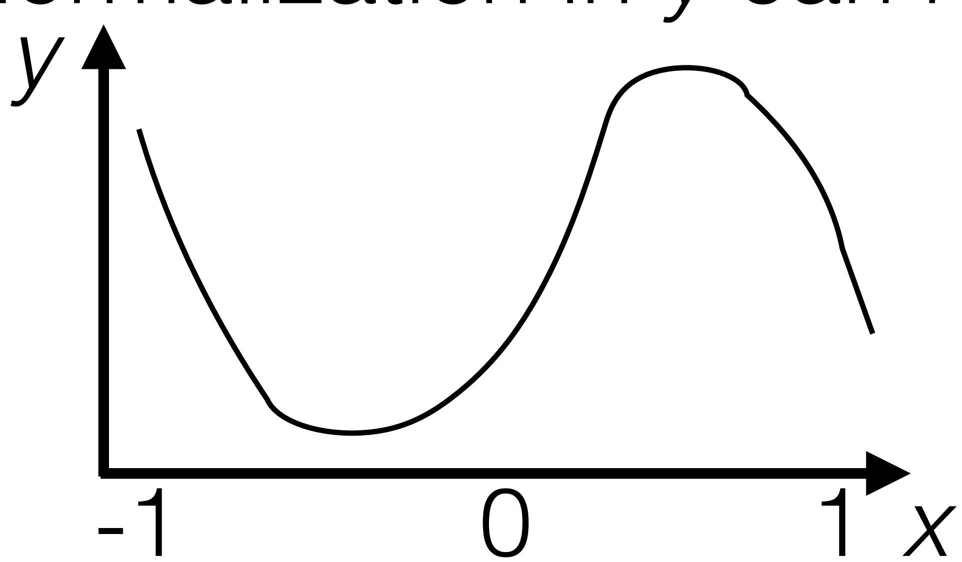
- What if we happened to measure our data on a different scale? [demo1, demo2]
- We've been using this particular kernel:

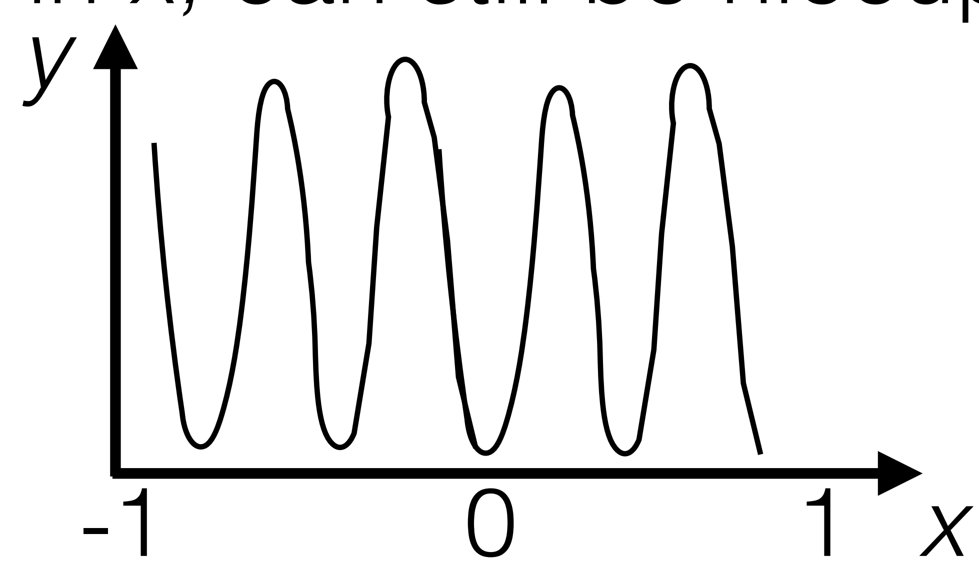
$$k(x, x') = \sigma^2 \exp(-\tfrac{1}{2}(x - x')^2), \sigma = 1$$

  - What do we expect from the scale of *f*(*x*) a priori?
    - At one *x*, with ~95% probability a priori, $f(x) \in (-2, 2)$
    - Marginal variance cannot increase with data
  - What counts as "close" in *x*?

$$\exp(-\tfrac{1}{2}2^2) \approx 0.14 \quad \exp(-\tfrac{1}{2}3^2) \approx 0.011 \quad \exp(-\tfrac{1}{2}4^2) \approx 0.00034$$

- What can we do to handle different *x* and *f*(*x*) scales?
  - Normalization in *y* can help; in *x*, can still be hiccups





Squared exponential kernel revisited

- A common option in practice and in software is to fit the *hyperparameters* of a more general *squared exponential kernel* from data

Squared exponential kernel revisited

- A common option in practice and in software is to fit the *hyperparameters* of a more general *squared exponential kernel* from data
  - More general form of the squared exponential:

$$k(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp(-\frac{1}{2} \sum_{d=1}^{D} \frac{(x_d - x'_d)^2}{\ell_d^2})$$

Squared exponential kernel revisited

- A common option in practice and in software is to fit the *hyperparameters* of a more general *squared exponential kernel* from data
  - More general form of the squared exponential:

$$k(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp(-\frac{1}{2} \sum_{d=1}^{D} \frac{(x_d - x'_d)^2}{\ell_d^2})$$

signal variance (σ^2)

lengthscales (ℓ_d^2)

Squared exponential kernel revisited

- A common option in practice and in software is to fit the *hyperparameters* of a more general *squared exponential kernel* from data
  - More general form of the squared exponential:

$$k(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp(-\frac{1}{2} \sum_{d=1}^{D} \frac{(x_d - x'_d)^2}{\ell_d^2})$$

signal variance lengthscales

  - *Parameters* (here, *f*) parametrize the distribution of the data. If we knew them, we could generate the data.

Squared exponential kernel revisited

- A common option in practice and in software is to fit the *hyperparameters* of a more general *squared exponential kernel* from data
  - More general form of the squared exponential:

$$k(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp(-\frac{1}{2} \sum_{d=1}^{D} \frac{(x_d - x'_d)^2}{\ell_d^2})$$

signal variance lengthscales

  - *Parameters* (here, *f*) parametrize the distribution of the data. If we knew them, we could generate the data.
    - GPs: *nonparametric* model: infinite # of latent params

Squared exponential kernel revisited

- A common option in practice and in software is to fit the *hyperparameters* of a more general *squared exponential kernel* from data
  - More general form of the squared exponential:

$$k(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp(-\frac{1}{2} \sum_{d=1}^{D} \frac{(x_d - x'_d)^2}{\ell_d^2})$$

signal variance   lengthscales

  - *Parameters* (here, *f*) parametrize the distribution of the data. If we knew them, we could generate the data.
    - GPs: *nonparametric* model: infinite # of latent params
  - *Hyperparameters* parametrize the distribution of the parameters. If known, we could generate the parameters.

Squared exponential kernel revisited

- A common option in practice and in software is to fit the *hyperparameters* of a more general *squared exponential kernel* from data
  - More general form of the squared exponential:

$$k(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp(-\frac{1}{2} \sum_{d=1}^{D} \frac{(x_d - x'_d)^2}{\ell_d^2})$$

signal variance lengthscales

  - *Parameters* (here, *f*) parametrize the distribution of the data. If we knew them, we could generate the data.
    - GPs: *nonparametric* model: infinite # of latent params
  - *Hyperparameters* parametrize the distribution of the parameters. If known, we could generate the parameters.
- Algorithm:

Squared exponential kernel revisited

- A common option in practice and in software is to fit the *hyperparameters* of a more general *squared exponential kernel* from data
  - More general form of the squared exponential:

$$k(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp(-\frac{1}{2} \sum_{d=1}^{D} \frac{(x_d - x'_d)^2}{\ell_d^2})$$

signal variance lengthscales

  - *Parameters* (here, *f*) parametrize the distribution of the data. If we knew them, we could generate the data.
    - GPs: *nonparametric* model: infinite # of latent params
  - *Hyperparameters* parametrize the distribution of the parameters. If known, we could generate the parameters.
- Algorithm:
  - Fit a value for the hyperparameters using the data.

Squared exponential kernel revisited

- A common option in practice and in software is to fit the *hyperparameters* of a more general *squared exponential kernel* from data
  - More general form of the squared exponential:

$$k(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp(-\frac{1}{2} \sum_{d=1}^{D} \frac{(x_d - x'_d)^2}{\ell_d^2})$$

signal variance lengthscales

  - *Parameters* (here, *f*) parametrize the distribution of the data. If we knew them, we could generate the data.
    - GPs: *nonparametric* model: infinite # of latent params
  - *Hyperparameters* parametrize the distribution of the parameters. If known, we could generate the parameters.
- Algorithm:
  - Fit a value for the hyperparameters using the data.
  - Given those values, now compute and report the mean and uncertainty intervals.

Squared exponential kernel revisited

- A common option in practice and in software is to fit the *hyperparameters* of a more general *squared exponential kernel* from data
  - More general form of the squared exponential:

$$k(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp\left(-\frac{1}{2} \sum_{d=1}^{D} \frac{(x_d - x'_d)^2}{\ell_d^2}\right)$$

signal variance　　　　lengthscales

  - *Parameters* (here, *f*) parametrize the distribution of the data. If we knew them, we could generate the data.
    - GPs: *nonparametric* model: infinite # of latent params
  - *Hyperparameters* parametrize the distribution of the parameters. If known, we could generate the parameters.
- Algorithm:
  - Fit a value for the hyperparameters using the data.
  - Given those values, now compute and report the mean and uncertainty intervals. [demo1,2,3]

Observation noise

Observation noise

- So far we've been assuming that we observed $f(x)$ directly

Observation noise

- So far we've been assuming that we observed $f(x)$ directly
- But often the actual observation y has additional noise:

Observation noise

- So far we've been assuming that we observed *f*(*x*) directly
- But often the actual observation *y* has additional noise:

$$f \sim \mathcal{GP}(m, k), y^{(n)} \sim f(\mathbf{x}^{(n)}) + \epsilon^{(n)}, \epsilon^{(n)} \stackrel{iid}{\sim} \mathcal{N}(0, \tau^2)$$

Observation noise

- So far we've been assuming that we observed *f*(*x*) directly
- But often the actual observation *y* has additional noise:

$$f \sim \mathcal{GP}(m, k), y^{(n)} \sim f(\mathbf{x}^{(n)}) + \epsilon^{(n)}, \epsilon^{(n)} \overset{iid}{\sim} \mathcal{N}(0, \tau^2)$$

  - We observe $\{(\mathbf{x}^{(n)}, y^{(n)})\}_{n=1}^N$ and want to learn the latent *f*

Observation noise

- So far we've been assuming that we observed *f*(*x*) directly
- But often the actual observation *y* has additional noise:

$$f \sim \mathcal{GP}(m, k), y^{(n)} \sim f(\mathbf{x}^{(n)}) + \epsilon^{(n)}, \epsilon^{(n)} \overset{iid}{\sim} \mathcal{N}(0, \tau^2)$$

  - We observe $\{(\mathbf{x}^{(n)}, y^{(n)})\}_{n=1}^{N}$ and want to learn the latent *f*

[demo1]

Observation noise

- So far we've been assuming that we observed *f*(*x*) directly
- But often the actual observation *y* has additional noise:

$$f \sim \mathcal{GP}(m, k), y^{(n)} \sim f(\mathbf{x}^{(n)}) + \epsilon^{(n)}, \epsilon^{(n)} \overset{iid}{\sim} \mathcal{N}(0, \tau^2)$$

  - We observe $\{(\mathbf{x}^{(n)}, y^{(n)})\}_{n=1}^{N}$ and want to learn the latent *f*
  - The *y*'s are multivariate-Gaussian-distributed

[demo1]

Why?

Observation noise

- So far we've been assuming that we observed *f*(*x*) directly
- But often the actual observation *y* has additional noise:

$$f \sim \mathcal{GP}(m, k), y^{(n)} \sim f(\mathbf{x}^{(n)}) + \epsilon^{(n)}, \epsilon^{(n)} \overset{iid}{\sim} \mathcal{N}(0, \tau^2)$$

  - We observe $\{(\mathbf{x}^{(n)}, y^{(n)})\}_{n=1}^{N}$ and want to learn the latent *f*
  - The *y*'s are multivariate-Gaussian-distributed [demo1]
    - Note: the sum of independent Gaussians is a Gaussian with means summed and covariances summed

Observation noise

- So far we've been assuming that we observed *f*(*x*) directly
- But often the actual observation *y* has additional noise:

$$f \sim \mathcal{GP}(m, k), y^{(n)} \sim f(\mathbf{x}^{(n)}) + \epsilon^{(n)}, \epsilon^{(n)} \overset{iid}{\sim} \mathcal{N}(0, \tau^2)$$

  - We observe $\{(\mathbf{x}^{(n)}, y^{(n)})\}_{n=1}^{N}$ and want to learn the latent *f*
  - The *y*'s are multivariate-Gaussian-distributed [demo1]
    - Note: the sum of independent Gaussians is a Gaussian with means summed and covariances summed
    - So the mean of $y^{(n)}$ is ?

Observation noise

- So far we've been assuming that we observed *f*(*x*) directly
- But often the actual observation *y* has additional noise:

$$f \sim \mathcal{GP}(m, k), y^{(n)} \sim f(\mathbf{x}^{(n)}) + \epsilon^{(n)}, \epsilon^{(n)} \overset{iid}{\sim} \mathcal{N}(0, \tau^2)$$

  - We observe $\{(\mathbf{x}^{(n)}, y^{(n)})\}_{n=1}^{N}$ and want to learn the latent *f*
  - The *y*'s are multivariate-Gaussian-distributed [demo1]
    - Note: the sum of independent Gaussians is a Gaussian with means summed and covariances summed
    - So the mean of $y^{(n)}$ is $m(\mathbf{x}^{(n)})$ and

Observation noise

- So far we've been assuming that we observed *f*(*x*) directly
- But often the actual observation *y* has additional noise:

$$f \sim \mathcal{GP}(m, k), y^{(n)} \sim f(\mathbf{x}^{(n)}) + \epsilon^{(n)}, \epsilon^{(n)} \overset{iid}{\sim} \mathcal{N}(0, \tau^2)$$

  - We observe $\{(\mathbf{x}^{(n)}, y^{(n)})\}_{n=1}^N$ and want to learn the latent *f*
  - The *y*'s are multivariate-Gaussian-distributed [demo1]
    - Note: the sum of independent Gaussians is a Gaussian with means summed and covariances summed
    - So the mean of $y^{(n)}$ is $m(\mathbf{x}^{(n)})$ and

$\mathrm{Cov}(y^{(n)}, y^{(n')}) =$?

Observation noise

- So far we've been assuming that we observed *f*(*x*) directly
- But often the actual observation *y* has additional noise:

$$f \sim \mathcal{GP}(m, k), y^{(n)} \sim f(\mathbf{x}^{(n)}) + \epsilon^{(n)}, \epsilon^{(n)} \stackrel{iid}{\sim} \mathcal{N}(0, \tau^2)$$

  - We observe $\{(\mathbf{x}^{(n)}, y^{(n)})\}_{n=1}^{N}$ and want to learn the latent *f*

[demo1]

  - The *y*'s are multivariate-Gaussian-distributed
    - Note: the sum of independent Gaussians is a Gaussian with means summed and covariances summed
    - So the mean of $y^{(n)}$ is $m(\mathbf{x}^{(n)})$ and

$$\mathrm{Cov}(y^{(n)}, y^{(n')}) = k(\mathbf{x}^{(n)}, \mathbf{x}^{(n')}) + \tau^2 \mathbf{1}\{n = n'\}$$

Observation noise

- So far we've been assuming that we observed *f*(*x*) directly
- But often the actual observation *y* has additional noise:

$$f \sim \mathcal{GP}(m, k), y^{(n)} \sim f(\mathbf{x}^{(n)}) + \epsilon^{(n)}, \epsilon^{(n)} \overset{iid}{\sim} \mathcal{N}(0, \tau^2)$$

  - We observe $\{(\mathbf{x}^{(n)}, y^{(n)})\}_{n=1}^{N}$ and want to learn the latent *f*

[demo1]

  - The *y*'s are multivariate-Gaussian-distributed
    - Note: the sum of independent Gaussians is a Gaussian with means summed and covariances summed
    - So the mean of $y^{(n)}$ is $m(\mathbf{x}^{(n)})$ and

$$\mathrm{Cov}(y^{(n)}, y^{(n')}) = k(\mathbf{x}^{(n)}, \mathbf{x}^{(n')}) + \tau^2 \mathbf{1}\{n = n'\}$$

Why compare indices, not *x*'s?

Observation noise

- So far we've been assuming that we observed *f*(*x*) directly
- But often the actual observation *y* has additional noise:

$$f \sim \mathcal{GP}(m, k), y^{(n)} \sim f(\mathbf{x}^{(n)}) + \epsilon^{(n)}, \epsilon^{(n)} \overset{iid}{\sim} \mathcal{N}(0, \tau^2)$$

  - We observe $\{(\mathbf{x}^{(n)}, y^{(n)})\}_{n=1}^N$ and want to learn the latent *f*

[demo1]

  - The *y*'s are multivariate-Gaussian-distributed
    - Note: the sum of independent Gaussians is a Gaussian with means summed and covariances summed
    - So the mean of $y^{(n)}$ is $m(\mathbf{x}^{(n)})$ and

$$\mathrm{Cov}(y^{(n)}, y^{(n')}) = k(\mathbf{x}^{(n)}, \mathbf{x}^{(n')}) + \tau^2 \mathbf{1}\{n = n'\}$$

Why compare indices, not *x*'s?

    - Before: $\begin{bmatrix} f(X) \\ f(X') \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(X, X) & K(X, X') \\ K(X', X) & K(X', X') \end{bmatrix}\right)$

Observation noise

- So far we've been assuming that we observed *f*(*x*) directly
- But often the actual observation *y* has additional noise:

$$f \sim \mathcal{GP}(m,k), y^{(n)} \sim f(\mathbf{x}^{(n)}) + \epsilon^{(n)}, \epsilon^{(n)} \stackrel{iid}{\sim} \mathcal{N}(0,\tau^2)$$

  - We observe $\{(\mathbf{x}^{(n)}, y^{(n)})\}_{n=1}^N$ and want to learn the latent *f*

[demo1]

  - The *y*'s are multivariate-Gaussian-distributed
    - Note: the sum of independent Gaussians is a Gaussian with means summed and covariances summed
    - So the mean of $y^{(n)}$ is $m(\mathbf{x}^{(n)})$ and

$$\mathrm{Cov}(y^{(n)}, y^{(n')}) = k(\mathbf{x}^{(n)}, \mathbf{x}^{(n')}) + \tau^2 \mathbf{1}\{n = n'\}$$

Why compare indices, not *x*'s?

- Before: $\begin{bmatrix} f(X) \\ f(X') \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(X,X) & K(X,X') \\ K(X',X) & K(X',X') \end{bmatrix}\right)$
- Now: $\begin{bmatrix} y^{(1:N)} \\ f(X') \end{bmatrix}$

Observation noise

- So far we've been assuming that we observed *f*(*x*) directly
- But often the actual observation *y* has additional noise:

$$f \sim \mathcal{GP}(m, k), y^{(n)} \sim f(\mathbf{x}^{(n)}) + \epsilon^{(n)}, \epsilon^{(n)} \overset{iid}{\sim} \mathcal{N}(0, \tau^2)$$

  - We observe $\{(\mathbf{x}^{(n)}, y^{(n)})\}_{n=1}^{N}$ and want to learn the latent *f*

[demo1]

  - The *y*'s are multivariate-Gaussian-distributed
    - Note: the sum of independent Gaussians is a Gaussian with means summed and covariances summed
    - So the mean of $y^{(n)}$ is $m(\mathbf{x}^{(n)})$ and

$$\mathrm{Cov}(y^{(n)}, y^{(n')}) = k(\mathbf{x}^{(n)}, \mathbf{x}^{(n')}) + \tau^2 \mathbf{1}\{n = n'\}$$

Why compare indices, not *x*'s?

- Before: $\begin{bmatrix} f(X) \\ f(X') \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(X,X) & K(X,X') \\ K(X',X) & K(X',X') \end{bmatrix}\right)$
- Now: $\begin{bmatrix} y^{(1:N)} \\ f(X') \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(X,X) + \tau^2 I & K(X,X') \\ K(X',X) & K(X',X') \end{bmatrix}\right)$

Observation noise

- So far we've been assuming that we observed *f*(*x*) directly
- But often the actual observation *y* has additional noise:

$$f \sim \mathcal{GP}(m, k), y^{(n)} \sim f(\mathbf{x}^{(n)}) + \epsilon^{(n)}, \epsilon^{(n)} \overset{iid}{\sim} \mathcal{N}(0, \tau^2)$$

  - We observe $\{(\mathbf{x}^{(n)}, y^{(n)})\}_{n=1}^{N}$ and want to learn the latent *f*
  - The *y*'s are multivariate-Gaussian-distributed [demo1]
    - Note: the sum of independent Gaussians is a Gaussian with means summed and covariances summed
    - So the mean of $y^{(n)}$ is $m(\mathbf{x}^{(n)})$ and

$$\mathrm{Cov}(y^{(n)}, y^{(n')}) = k(\mathbf{x}^{(n)}, \mathbf{x}^{(n')}) + \tau^2 \mathbf{1}\{n = n'\}$$

Why compare indices, not *x*'s?

- Before: $\begin{bmatrix} f(X) \\ f(X') \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(X,X) & K(X,X') \\ K(X',X) & K(X',X') \end{bmatrix}\right)$
- Now: $\begin{bmatrix} y^{(1:N)} \\ f(X') \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(X,X) + \tau^2 I & K(X,X') \\ K(X',X) & K(X',X') \end{bmatrix}\right)$

Observation noise

- So far we've been assuming that we observed *f*(*x*) directly
- But often the actual observation *y* has additional noise:

$$f \sim \mathcal{GP}(m, k), y^{(n)} \sim f(\mathbf{x}^{(n)}) + \epsilon^{(n)}, \epsilon^{(n)} \overset{iid}{\sim} \mathcal{N}(0, \tau^2)$$

  - We observe $\{(\mathbf{x}^{(n)}, y^{(n)})\}_{n=1}^{N}$ and want to learn the latent *f*
  - The *y*'s are multivariate-Gaussian-distributed [demo1]
    - Note: the sum of independent Gaussians is a Gaussian with means summed and covariances summed
    - So the mean of $y^{(n)}$ is $m(\mathbf{x}^{(n)})$ and

$$\mathrm{Cov}(y^{(n)}, y^{(n')}) = k(\mathbf{x}^{(n)}, \mathbf{x}^{(n')}) + \tau^2 \mathbf{1}\{n = n'\}$$

Why compare indices, not *x*'s?

- Before: $\begin{bmatrix} f(X) \\ f(X') \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(X,X) & K(X,X') \\ K(X',X) & K(X',X') \end{bmatrix}\right)$
- Now: $\begin{bmatrix} y^{(1:N)} \\ f(X') \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(X,X) + \tau^2 I & K(X,X') \\ K(X',X) & K(X',X') \end{bmatrix}\right)$

Observation noise

- So far we've been assuming that we observed *f*(*x*) directly
- But often the actual observation *y* has additional noise:

$$f \sim \mathcal{GP}(m, k), y^{(n)} \sim f(\mathbf{x}^{(n)}) + \epsilon^{(n)}, \epsilon^{(n)} \overset{iid}{\sim} \mathcal{N}(0, \tau^2)$$

  - We observe $\{(\mathbf{x}^{(n)}, y^{(n)})\}_{n=1}^N$ and want to learn the latent *f*

[demo1]

  - The *y*'s are multivariate-Gaussian-distributed
    - Note: the sum of independent Gaussians is a Gaussian with means summed and covariances summed
    - So the mean of $y^{(n)}$ is $m(\mathbf{x}^{(n)})$ and

$$\mathrm{Cov}(y^{(n)}, y^{(n')}) = k(\mathbf{x}^{(n)}, \mathbf{x}^{(n')}) + \tau^2 \mathbf{1}\{n = n'\}$$

Why compare indices, not *x*'s?

- Before: $\begin{bmatrix} f(X) \\ f(X') \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(X, X) & K(X, X') \\ K(X', X) & K(X', X') \end{bmatrix}\right)$
- Now: $\begin{bmatrix} y^{(1:N)} \\ f(X') \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(X, X) + \tau^2 I & K(X, X') \\ K(X', X) & K(X', X') \end{bmatrix}\right)$

What if we put *y* here instead?

Observation noise

- So far we've been assuming that we observed *f*(*x*) directly
- But often the actual observation *y* has additional noise:

$$f \sim \mathcal{GP}(m, k), y^{(n)} \sim f(\mathbf{x}^{(n)}) + \epsilon^{(n)}, \epsilon^{(n)} \overset{iid}{\sim} \mathcal{N}(0, \tau^2)$$

  - We observe $\{(\mathbf{x}^{(n)}, y^{(n)})\}_{n=1}^{N}$ and want to learn the latent *f*
  - The *y*'s are multivariate-Gaussian-distributed [demo1]
    - Note: the sum of independent Gaussians is a Gaussian with means summed and covariances summed
    - So the mean of $y^{(n)}$ is $m(\mathbf{x}^{(n)})$ and

$$\mathrm{Cov}(y^{(n)}, y^{(n')}) = k(\mathbf{x}^{(n)}, \mathbf{x}^{(n')}) + \tau^2 \mathbf{1}\{n = n'\}$$

Why compare indices, not *x*'s?

- Before: $\begin{bmatrix} f(X) \\ f(X') \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(X,X) & K(X,X') \\ K(X',X) & K(X',X') \end{bmatrix}\right)$
- Now: $\begin{bmatrix} y^{(1:N)} \\ f(X') \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(X,X) + \tau^2 I & K(X,X') \\ K(X',X) & K(X',X') \end{bmatrix}\right)$

[demo2, demo3]

Observation noise

- So far we've been assuming that we observed *f*(*x*) directly
- But often the actual observation *y* has additional noise:

$$f \sim \mathcal{GP}(m, k), y^{(n)} \sim f(\mathbf{x}^{(n)}) + \epsilon^{(n)}, \epsilon^{(n)} \overset{iid}{\sim} \mathcal{N}(0, \tau^2)$$

  - We observe $\{(\mathbf{x}^{(n)}, y^{(n)})\}_{n=1}^{N}$ and want to learn the latent *f*
  - The *y*'s are multivariate-Gaussian-distributed [demo1]
    - Note: the sum of independent Gaussians is a Gaussian with means summed and covariances summed
    - So the mean of $y^{(n)}$ is $m(\mathbf{x}^{(n)})$ and

$$\mathrm{Cov}(y^{(n)}, y^{(n')}) = k(\mathbf{x}^{(n)}, \mathbf{x}^{(n')}) + \tau^2 \mathbf{1}\{n = n'\}$$

Why compare indices, not *x*'s?

- Before: $\begin{bmatrix} f(X) \\ f(X') \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(X,X) & K(X,X') \\ K(X',X) & K(X',X') \end{bmatrix}\right)$
- Now: $\begin{bmatrix} y^{(1:N)} \\ f(X') \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(X,X) + \tau^2 I & K(X,X') \\ K(X',X) & K(X',X') \end{bmatrix}\right)$

Can you state a non-trivial lower bound on the marginal variance of a test $y^{(m)}$?

[demo2, demo3]

Observation noise

Even when observations are "perfect," use a (very small) *nugget* for numerical reasons

- So far we've been assuming that we observed *f*(*x*) directly
- But often the actual observation *y* has additional noise:

$$f \sim \mathcal{GP}(m, k), y^{(n)} \sim f(\mathbf{x}^{(n)}) + \epsilon^{(n)}, \epsilon^{(n)} \overset{iid}{\sim} \mathcal{N}(0, \tau^2)$$

  - We observe $\{(\mathbf{x}^{(n)}, y^{(n)})\}_{n=1}^N$ and want to learn the latent *f*
  - The *y*'s are multivariate-Gaussian-distributed [demo1]
    - Note: the sum of independent Gaussians is a Gaussian with means summed and covariances summed
    - So the mean of $y^{(n)}$ is $m(\mathbf{x}^{(n)})$ and

$$\mathrm{Cov}(y^{(n)}, y^{(n')}) = k(\mathbf{x}^{(n)}, \mathbf{x}^{(n')}) + \tau^2 \mathbf{1}\{n = n'\}$$

Why compare indices, not *x*'s?

- Before: $\begin{bmatrix} f(X) \\ f(X') \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(X,X) & K(X,X') \\ K(X',X) & K(X',X') \end{bmatrix}\right)$
- Now: $\begin{bmatrix} y^{(1:N)} \\ f(X') \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(X,X) + \tau^2 I & K(X,X') \\ K(X',X) & K(X',X') \end{bmatrix}\right)$

Can you state a non-trivial lower bound on the marginal variance of a test $y^{(m)}$? [demo2, demo3]