Course in Bayesian Optimization

Javier González
(most of slides today: Neil Lawrence)

University of Sheffield, Sheffield, UK

27th October 2015

Many thanks to

- Mauricio Álvarez.
- Cristian Guarizno.
- Neil Lawrence, University of Sheffield.
- Zhenwen Dai, University of Sheffield.
- Machine Learning group, University of Sheffield.
- Philipp Hennig, Max Planck institute.
- Michael Osborne, University of Oxford.

Outline of the Course

- Lecture 1: Uncertainty and Gaussian Processes.
- Lecture 2: Introduction to Bayesian (probabilistic) optimization.
- Lecture 3: Advanced topics in Bayesian Optimization.

Outline of the Course

- Lab 1: Introduction to GPy.
- Lab 2: Introduction to GPyOpt.
- Lab 3: Advanced GPyOpt.
- Day 4: **Projects + presentations.**

Points of the day

What is machine learning?

What is the uncertainty? Types?

How the uncertainty plays a role in the learning process.

Gaussian processes as models to handle uncertainty.

Points of the day

What is machine learning?

What is the uncertainty? Types?

How the uncertainty plays a role in the learning process.

Gaussian processes as models to handle uncertainty.

Points of the day

What is machine learning?

What is the uncertainty? Types?

How the uncertainty plays a role in the learning process.

Gaussian processes as models to handle uncertainty.

Points of the day

What is machine learning?

What is the uncertainty? Types?

How the uncertainty plays a role in the learning process.

Gaussian processes as models to handle uncertainty.

Points of the day

What is machine learning?

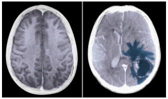
What is the uncertainty? Types?

How the uncertainty plays a role in the learning process.

Gaussian processes as models to handle uncertainty.

What is to learn?

The human learning process?



Cancerous Tumor?

What is Machine Learning?

data

- data: observations, could be actively or passively acquired (meta-data).
- model: assumptions, based on previous experience (other data! transfer learning etc), or beliefs about the regularities of the universe. Inductive bias.
- prediction: an action to be taken or a categorization or a quality score.

What is Machine Learning?

data +

- data: observations, could be actively or passively acquired (meta-data).
- model: assumptions, based on previous experience (other data! transfer learning etc), or beliefs about the regularities of the universe. Inductive bias.
- prediction: an action to be taken or a categorization or a quality score.

What is Machine Learning?

data + model

- data: observations, could be actively or passively acquired (meta-data).
- model: assumptions, based on previous experience (other data! transfer learning etc), or beliefs about the regularities of the universe. Inductive bias.
- prediction: an action to be taken or a categorization or a quality score.

What is Machine Learning?

data + model =

- data: observations, could be actively or passively acquired (meta-data).
- model: assumptions, based on previous experience (other data! transfer learning etc), or beliefs about the regularities of the universe. Inductive bias.
- prediction: an action to be taken or a categorization or a quality score.

What is Machine Learning?

data + model = prediction

- data: observations, could be actively or passively acquired (meta-data).
- model: assumptions, based on previous experience (other data! transfer learning etc), or beliefs about the regularities of the universe. Inductive bias.
- prediction: an action to be taken or a categorization or a quality score.

Historical Perspective

- A data driven approach to Artificial Intelligence.
- Inspired by attempts to model the brain (the connectionists).
- A community that transcended traditional boundaries (psychology, statistical physics, signal processing)
- Led to an approach that dominates in the modern data-rich world.

Two Dominant Approaches

- Machine Learning as Optimization:
  - Formulate your learning Problem as an optimization problem.
  - Typically intractable, so minimize a *relaxed* version of the cost function.
  - Prove characteristics of the resulting solution.
- Machine Learning as Probabilistic Modelling:
  - Formulate your learning problem as a probabilistic model.
  - Relate variables through probability distributions.
  - If *Bayesian*, treat parameters with probability distributions.
  - Required integrals often intractable: use approximations (MCMC, variational etc).

Two Dominant Approaches

- Machine Learning as Optimization:
  - Formulate your learning Problem as an optimization problem.
  - Typically intractable, so minimize a *relaxed* version of the cost function.
  - Prove characteristics of the resulting solution.
- Machine Learning as Probabilistic Modelling:
  - Formulate your learning problem as a probabilistic model.
  - Relate variables through probability distributions.
  - If *Bayesian*, treat parameters with probability distributions.
  - Required integrals often intractable: use approximations (MCMC, variational etc).

Two Dominant Approaches

- Machine Learning as Optimization:
  - Formulate your learning Problem as an optimization problem.
  - Typically intractable, so minimize a *relaxed* version of the cost function.
  - Prove characteristics of the resulting solution.
- Machine Learning as Probabilistic Modelling:
  - Formulate your learning problem as a probabilistic model.
  - Relate variables through probability distributions.
  - If *Bayesian*, treat parameters with probability distributions.
  - Required integrals often intractable: use approximations (MCMC, variational etc).

Two Dominant Approaches

- Machine Learning as Optimization:
  - Formulate your learning Problem as an optimization problem.
  - Typically intractable, so minimize a *relaxed* version of the cost function.
  - Prove characteristics of the resulting solution.
- Machine Learning as Probabilistic Modelling:
  - Formulate your learning problem as a probabilistic model.
  - Relate variables through probability distributions.
  - If *Bayesian*, treat parameters with probability distributions.
  - Required integrals often intractable: use approximations (MCMC, variational etc).

Two Dominant Approaches

- Machine Learning as Optimization:
  - Formulate your learning Problem as an optimization problem.
  - Typically intractable, so minimize a *relaxed* version of the cost function.
  - Prove characteristics of the resulting solution.
- Machine Learning as Probabilistic Modelling:
  - Formulate your learning problem as a probabilistic model.
  - Relate variables through probability distributions.
  - If *Bayesian*, treat parameters with probability distributions.
  - Required integrals often intractable: use approximations (MCMC, variational etc).

Two Dominant Approaches

- Machine Learning as Optimization:
  - Formulate your learning Problem as an optimization problem.
  - Typically intractable, so minimize a *relaxed* version of the cost function.
  - Prove characteristics of the resulting solution.
- Machine Learning as Probabilistic Modelling:
  - Formulate your learning problem as a probabilistic model.
  - Relate variables through probability distributions.
  - If *Bayesian*, treat parameters with probability distributions.
  - Required integrals often intractable: use approximations (MCMC, variational etc).

Two Dominant Approaches

- Machine Learning as Optimization:
  - Formulate your learning Problem as an optimization problem.
  - Typically intractable, so minimize a *relaxed* version of the cost function.
  - Prove characteristics of the resulting solution.
- Machine Learning as Probabilistic Modelling:
  - Formulate your learning problem as a probabilistic model.
  - Relate variables through probability distributions.
  - If *Bayesian*, treat parameters with probability distributions.
  - Required integrals often intractable: use approximations (MCMC, variational etc).

Two Dominant Approaches

- Machine Learning as Optimization:
  - Formulate your learning Problem as an optimization problem.
  - Typically intractable, so minimize a *relaxed* version of the cost function.
  - Prove characteristics of the resulting solution.
- Machine Learning as Probabilistic Modelling:
  - Formulate your learning problem as a probabilistic model.
  - Relate variables through probability distributions.
  - If *Bayesian*, treat parameters with probability distributions.
  - Required integrals often intractable: use approximations (MCMC, variational etc).

Two Dominant Approaches

- Machine Learning as Optimization:
  - Formulate your learning Problem as an optimization problem.
  - Typically intractable, so minimize a *relaxed* version of the cost function.
  - Prove characteristics of the resulting solution.
- Machine Learning as Probabilistic Modelling:
  - Formulate your learning problem as a probabilistic model.
  - Relate variables through probability distributions.
  - If *Bayesian*, treat parameters with probability distributions.
  - Required integrals often intractable: use approximations (MCMC, variational etc).

Modelling Assumptions

- Modelling assumptions are either included as:
  - a regularizer (optimization) or
  - in the probability distribution (probabilistic approach).
- Typical assumptions: sparsity, smoothness.

Modelling Assumptions

- Modelling assumptions are either included as:
  - a regularizer (optimization) or
  - in the probability distribution (probabilistic approach).
- Typical assumptions: sparsity, smoothness.

Modelling Assumptions

- Modelling assumptions are either included as:
  - a regularizer (optimization) or
  - in the probability distribution (probabilistic approach).
- Typical assumptions: sparsity, smoothness.

Modelling Assumptions

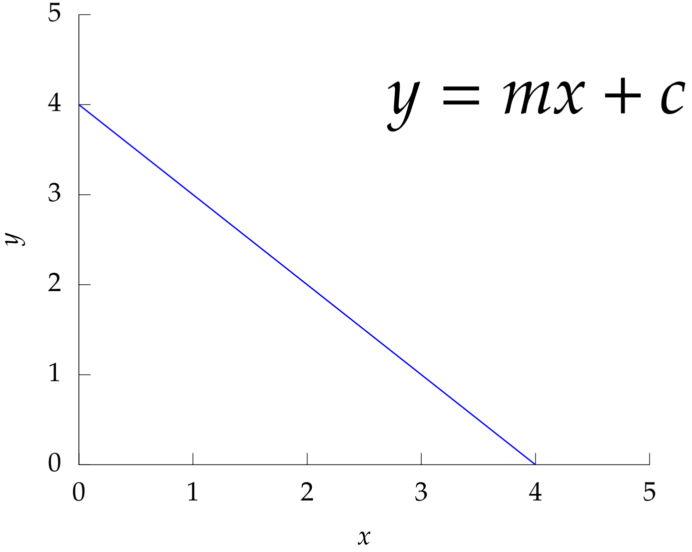
- Modelling assumptions are either included as:
  - a regularizer (optimization) or
  - in the probability distribution (probabilistic approach).
- Typical assumptions: sparsity, smoothness.

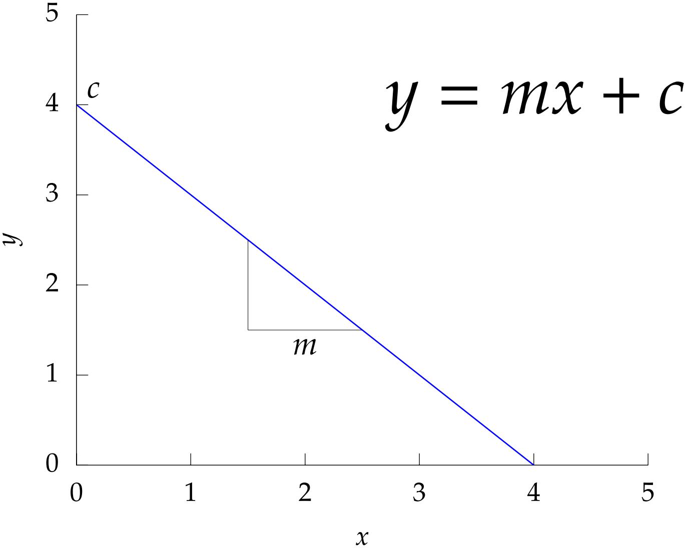
Applications of Machine Learning

- **Handwriting Recognition** : Recognising handwritten characters. For example LeNet `http://bit.ly/d26fwK`.
- **Friend Indentification** : Suggesting friends on social networks `https://www.facebook.com/help/501283333222485`
- **Ranking** : Learning relative skills of on line game players, the TrueSkill system `http://research.microsoft.com/en-us/projects/trueskill/`. `http://www.netflixprize.com/`.
- **Internet Search** : For example Ad Click Through rate prediction `http://bit.ly/a7XLH4`.
- **News Personalisation** : For example Zite `http://www.zite.com/`.
- **Game Play Learning** : For example, learning to play Go `http://bit.ly/cV77zM`.

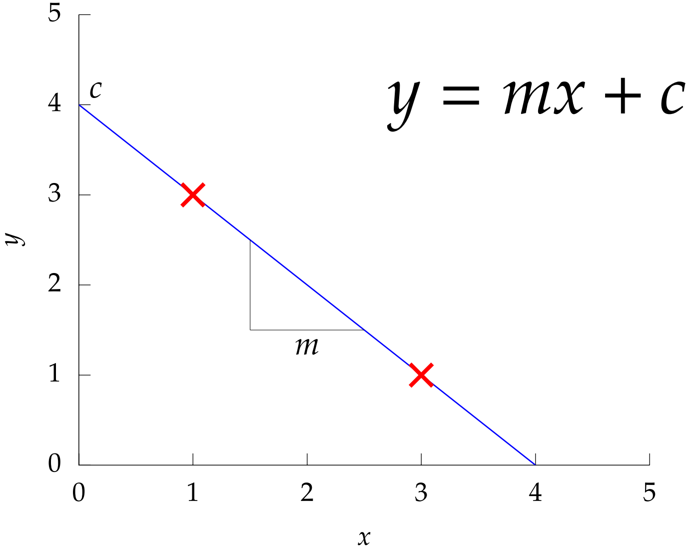
Learning is Optimization

$$y = mx + c$$

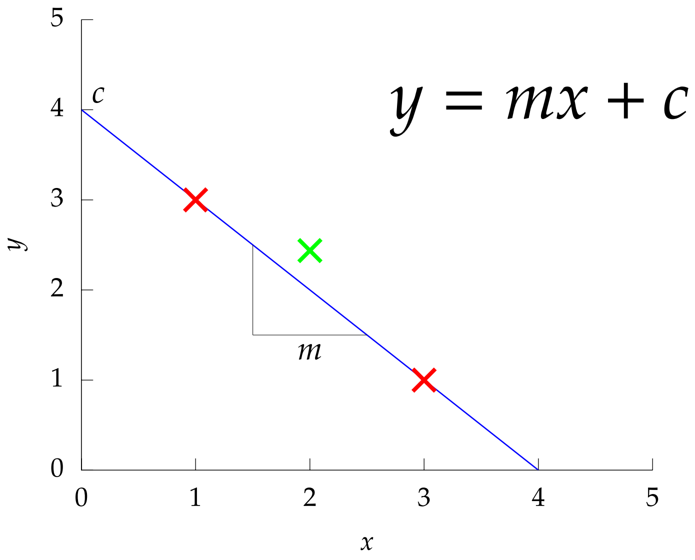

$y = mx + c$
5
4
3
2
1
0
y
0
1
2
3
4
5
x



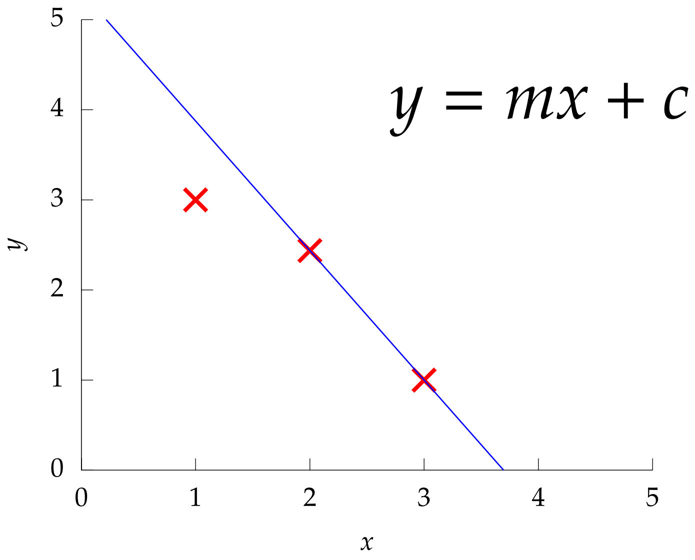
$y = mx + c$
c
m
y
x
0
1
2
3
4
5



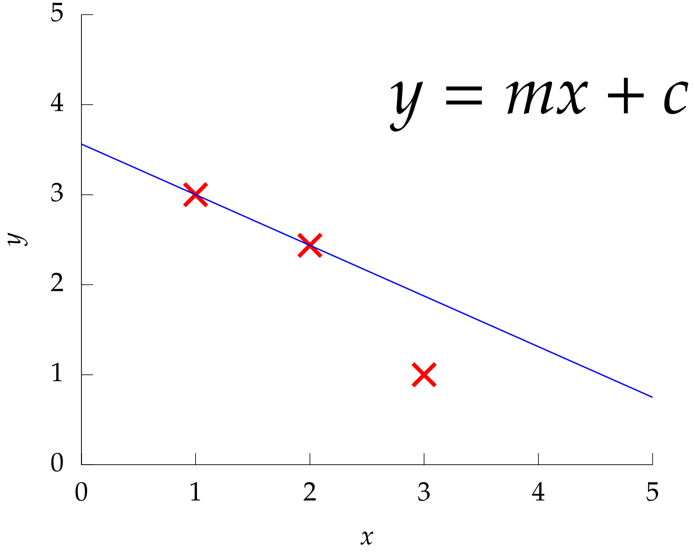
$y = mx + c$
c
m
0
1
2
3
4
5
0
1
2
3
4
5
x
y



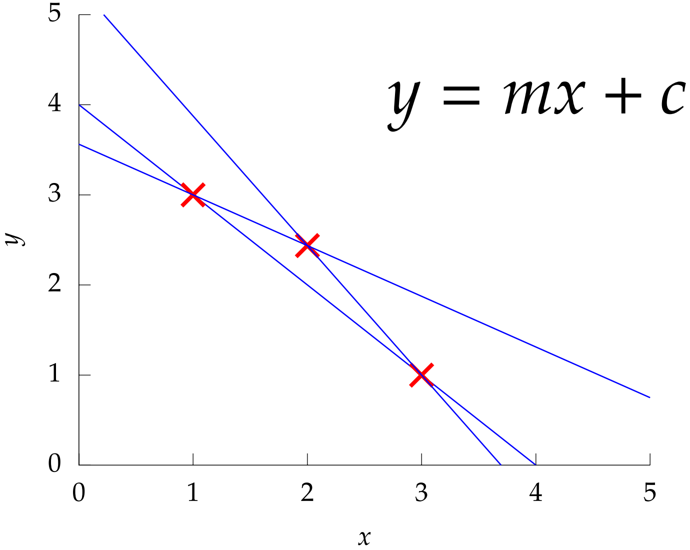
$y = mx + c$
c
m
y
x
0
1
2
3
4
5



$y = mx + c$
0
1
2
3
4
5
0
1
2
3
4
5
x
y



$y = mx + c$
y
x
0
1
2
3
4
5
0
1
2
3
4
5



$y = mx + c$
0
1
2
3
4
5
x
y

point 1: $x = 1, y = 3$

$$3 = m + c$$

point 2: $x = 3, y = 1$

$$1 = 3m + c$$

point 3: $x = 2, y = 2.5$

$$2.5 = 2m + c$$

point 1: $x = 1, y = 3$

$$3 = m + c + \epsilon_1$$

point 2: $x = 3, y = 1$

$$1 = 3m + c + \epsilon_2$$

point 3: $x = 2, y = 2.5$

$$2.5 = 2m + c + \epsilon_3$$

APPENDICE.

Sur la Méthode des moindres quarrés.

DANS la plupart des questions où il s'agit de tirer des mesures données par l'observation, les résultats les plus exacts qu'elles peuvent offrir, on est presque toujours conduit à un système d'équations de la forme

$$E = a + bx + cy + fz + \&c.$$

dans lesquelles a, b, c, f, &c. sont des coëfficiens connus, qui varient d'une équation à l'autre, et x, y, z, &c. sont des inconnues qu'il faut déterminer par la condition que la valeur de E se réduise, pour chaque équation, à une quantité ou nulle ou très-petite.

Si l'on a autant d'équations que d'inconnues x, y, z, &c., il n'y a aucune difficulté pour la détermination de ces inconnues, et on peut rendre les erreurs E absolument nulles. Mais le plus souvent, le nombre des équations est supérieur à celui des inconnues, et il est impossible d'anéantir toutes les erreurs.

Dans cette circonstance, qui est celle de la plupart des problêmes physiques et astronomiques, où l'on cherche à déter-

Regression Revisited

- We introduce an error function of the form

$$E(\mathbf{w}) = \sum_{i=1}^{n} (y_i - mx_i - c)^2$$

- Minimize the error function with respect to m and c

Mathematical Interpretation

- What is the mathematical interpretation?
  - There is a cost function.
  - It expresses mismatch between your prediction and reality.

$$E(m, c) = \sum_{i=1}^{n} (y_i - mx_i - c)^2$$

  - This is known as the sum of squares error.

Learning is Optimization

- Learning is minimization of the cost function.
- At the minima the gradient is zero.
- Coordinate ascent, find gradient in each coordinate and set to zero.

$$\frac{\mathrm{d}E(m)}{\mathrm{d}m} = -2\sum_{i=1}^{n} x_i \left(y_i - mx_i - c\right)$$

Learning is Optimization

- Learning is minimization of the cost function.
- At the minima the gradient is zero.
- Coordinate ascent, find gradient in each coordinate and set to zero.

$$0 = -2 \sum_{i=1}^{n} x_i \left(y_i - m x_i - c\right)$$

Learning is Optimization

- Learning is minimization of the cost function.
- At the minima the gradient is zero.
- Coordinate ascent, find gradient in each coordinate and set to zero.

$$0 = -2\sum_{i=1}^{n} x_i y_i + 2\sum_{i=1}^{n} m x_i^2 + 2\sum_{i=1}^{n} c x_i$$

Learning is Optimization

- Learning is minimization of the cost function.
- At the minima the gradient is zero.
- Coordinate ascent, find gradient in each coordinate and set to zero.

$$m = \frac{\sum_{i=1}^{n} (y_i - c) x_i}{\sum_{i=1}^{n} x_i^2}$$

Learning is Optimization

- Learning is minimization of the cost function.
- At the minima the gradient is zero.
- Coordinate ascent, find gradient in each coordinate and set to zero.

$$\frac{\mathrm{d}E(c)}{\mathrm{d}c} = -2\sum_{i=1}^{n}(y_i - mx_i - c)$$

Learning is Optimization

- Learning is minimization of the cost function.
- At the minima the gradient is zero.
- Coordinate ascent, find gradient in each coordinate and set to zero.

$$0 = -2 \sum_{i=1}^{n} (y_i - mx_i - c)$$

Learning is Optimization

- Learning is minimization of the cost function.
- At the minima the gradient is zero.
- Coordinate ascent, find gradient in each coordinate and set to zero.

$$0 = -2\sum_{i=1}^{n} y_i + 2\sum_{i=1}^{n} mx_i + 2nc$$

Learning is Optimization

- Learning is minimization of the cost function.
- At the minima the gradient is zero.
- Coordinate ascent, find gradient in each coordinate and set to zero.

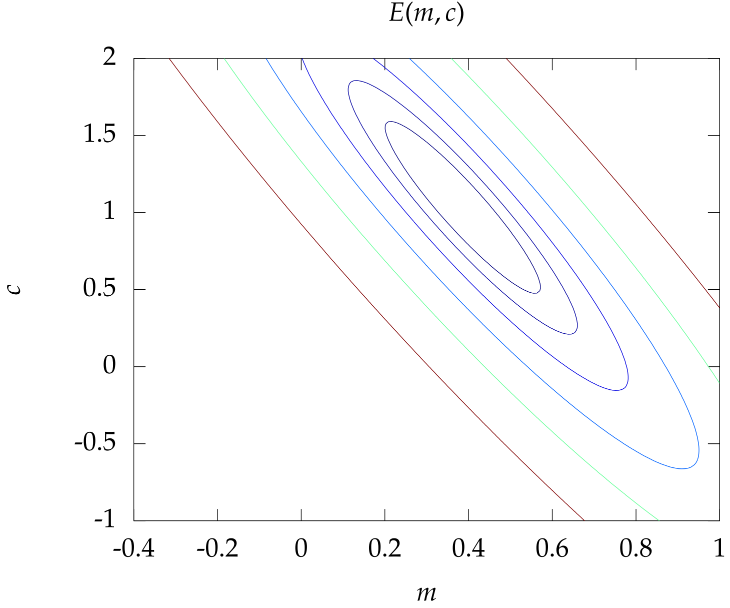
$$c = \frac{\sum_{i=1}^{n} (y_i - mx_i)}{n}$$

Fixed Point Updates

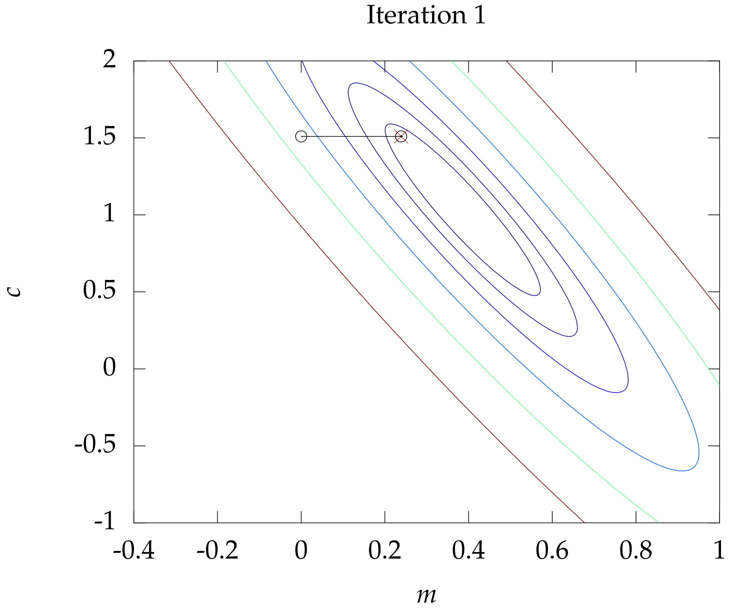
Worked example.

$$
\begin{aligned}
c^* &= \frac{\sum_{i=1}^{n} (y_i - m^* x_i)}{n}, \\
m^* &= \frac{\sum_{i=1}^{n} x_i (y_i - c^*)}{\sum_{i=1}^{n} x_i^2}, \\
\sigma^{2\,*} &= \frac{\sum_{i=1}^{n} (y_i - m^* x_i - c^*)^2}{n}
\end{aligned}
$$

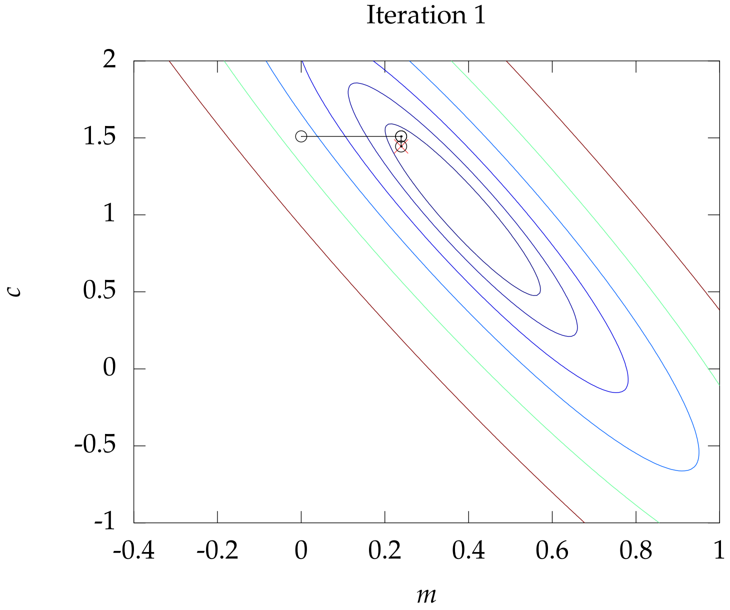
Coordinate Descent



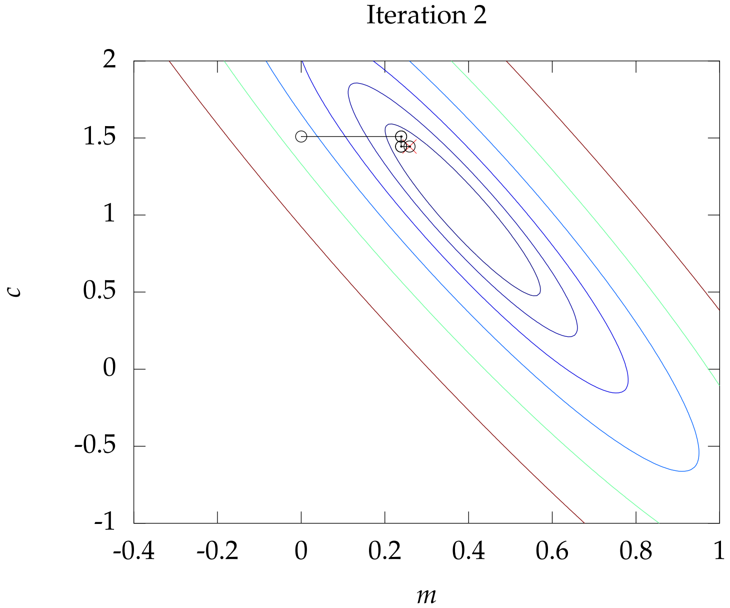
Coordinate Descent



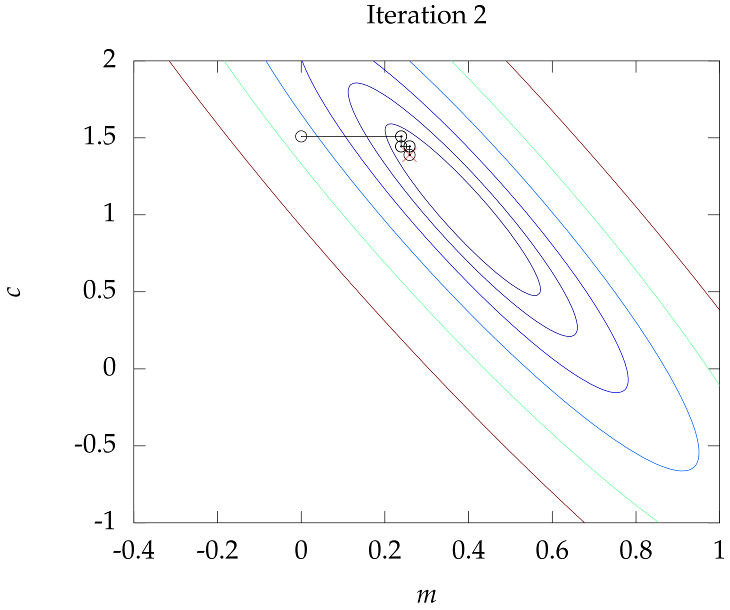
Coordinate Descent



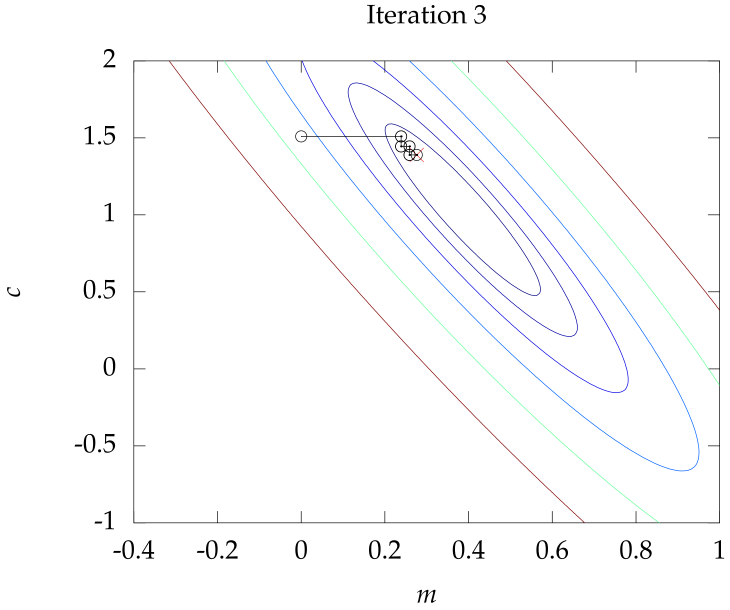
Coordinate Descent



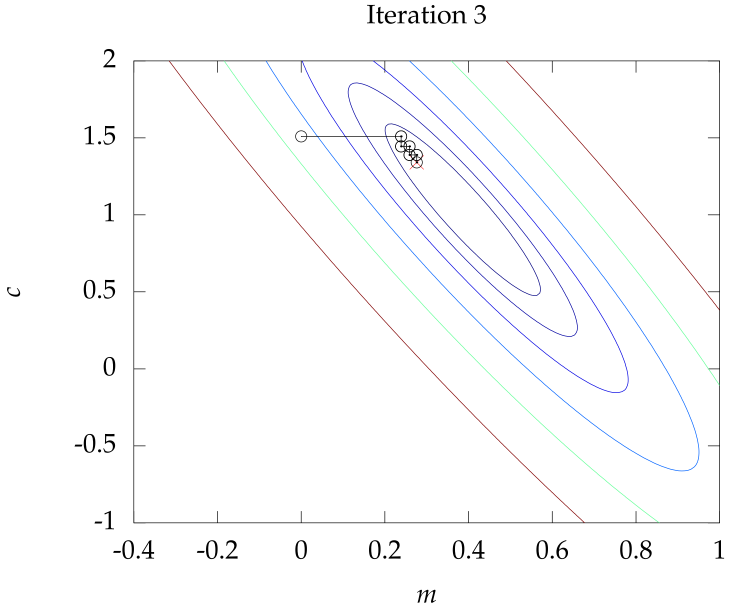
Coordinate Descent



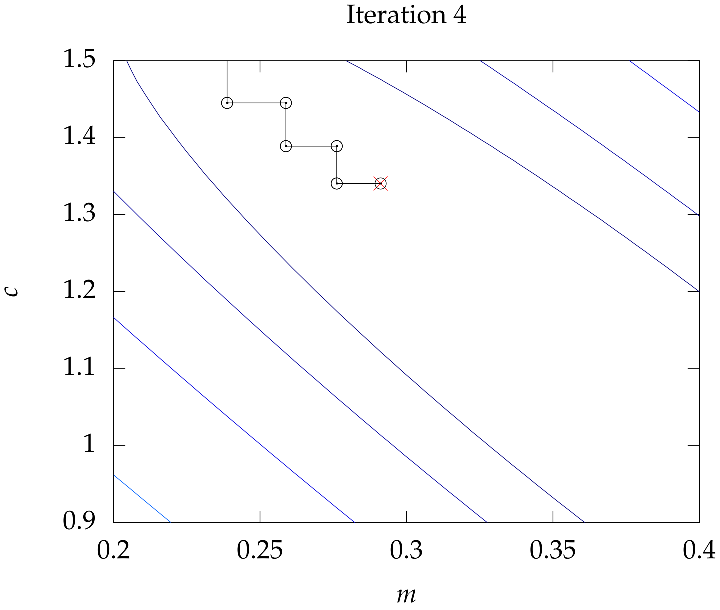
Coordinate Descent



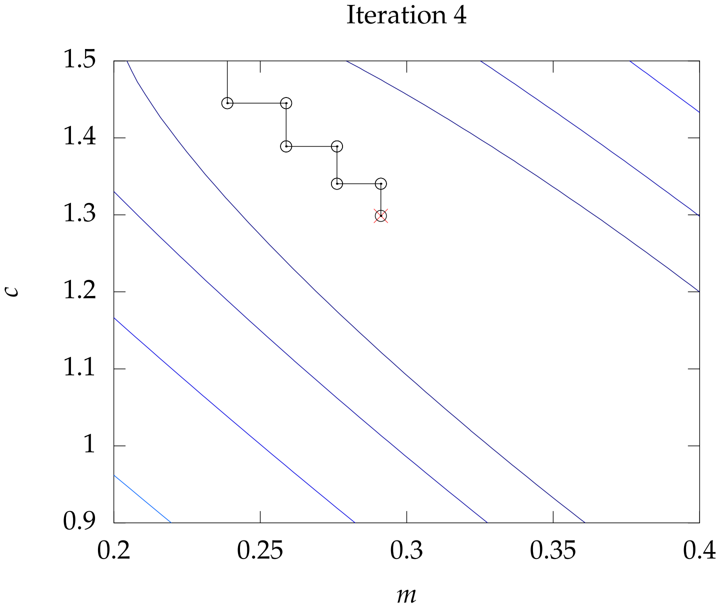
Coordinate Descent



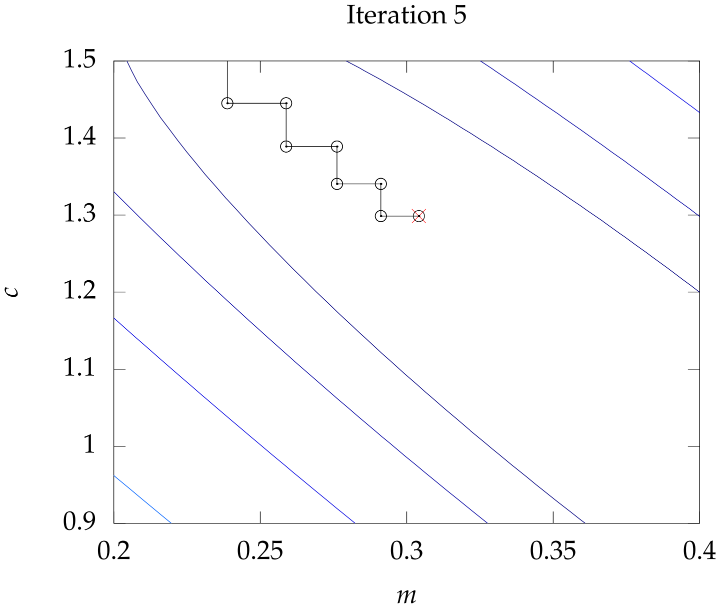
Coordinate Descent



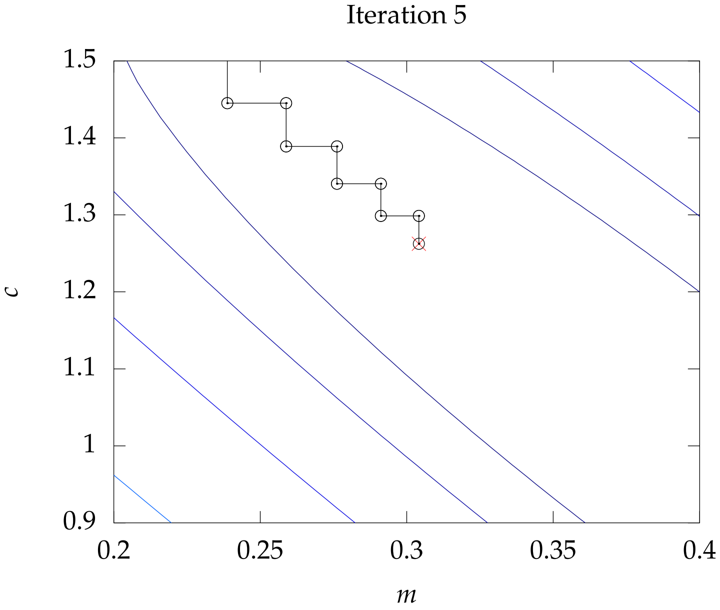
Coordinate Descent



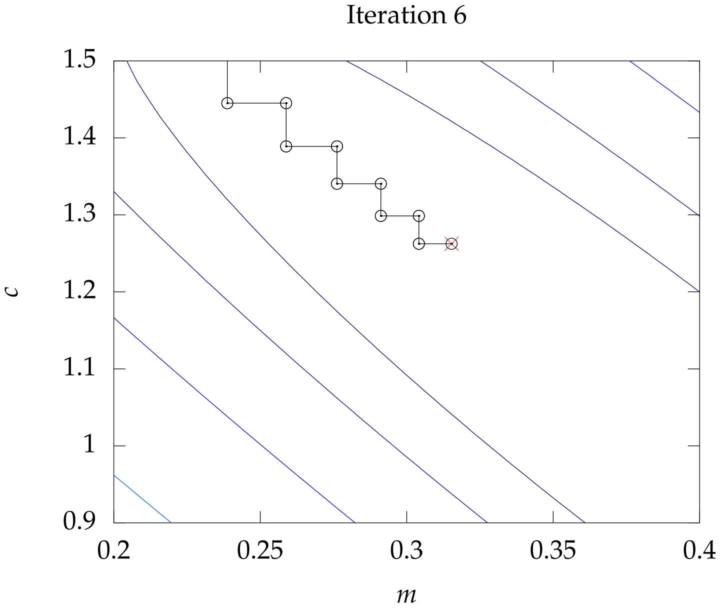
Coordinate Descent



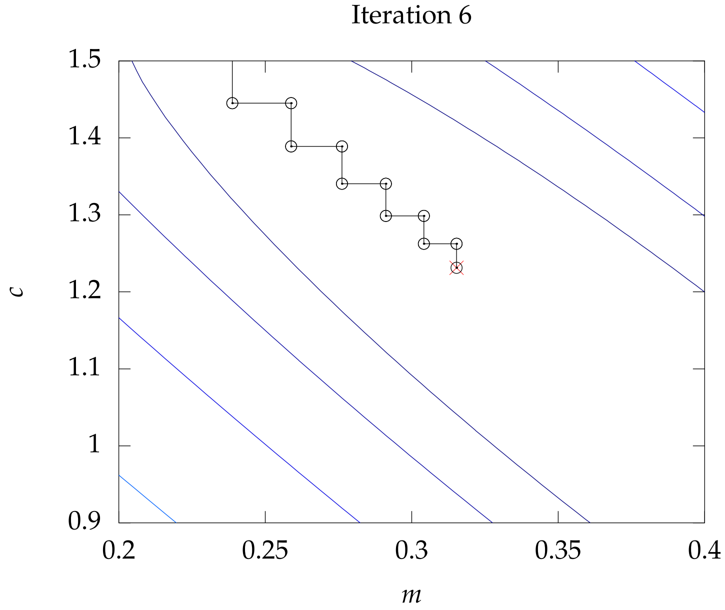
Coordinate Descent



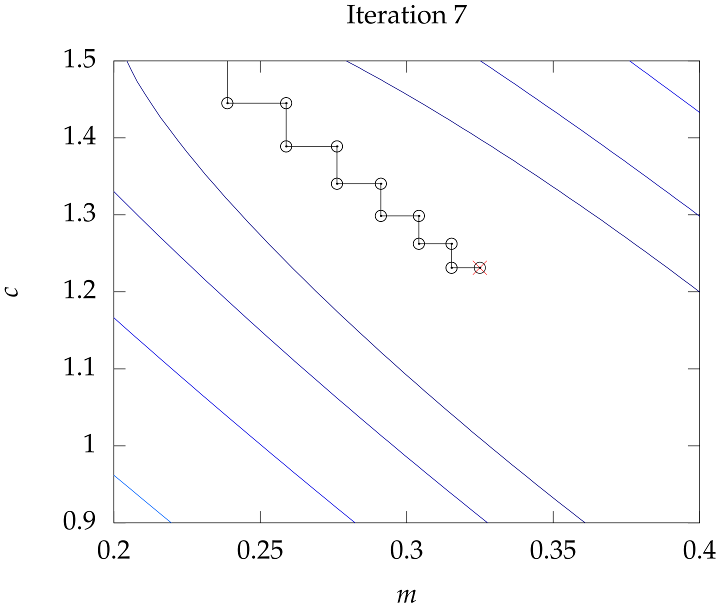
Coordinate Descent



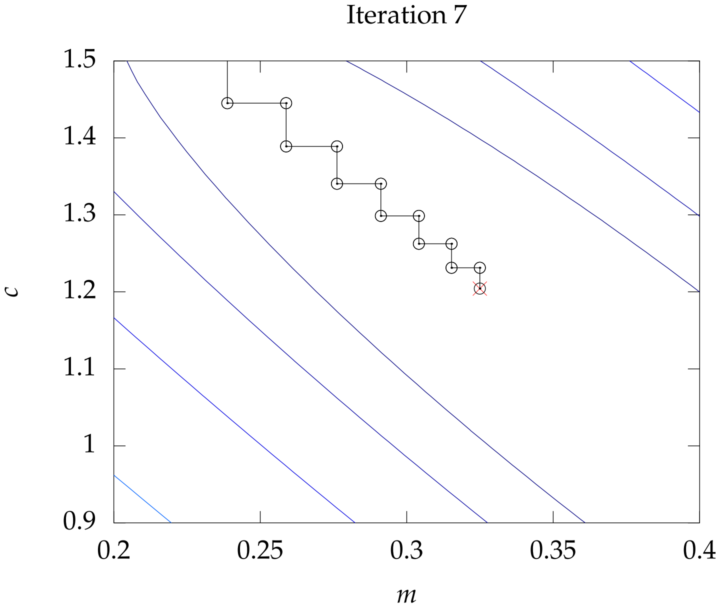
Coordinate Descent



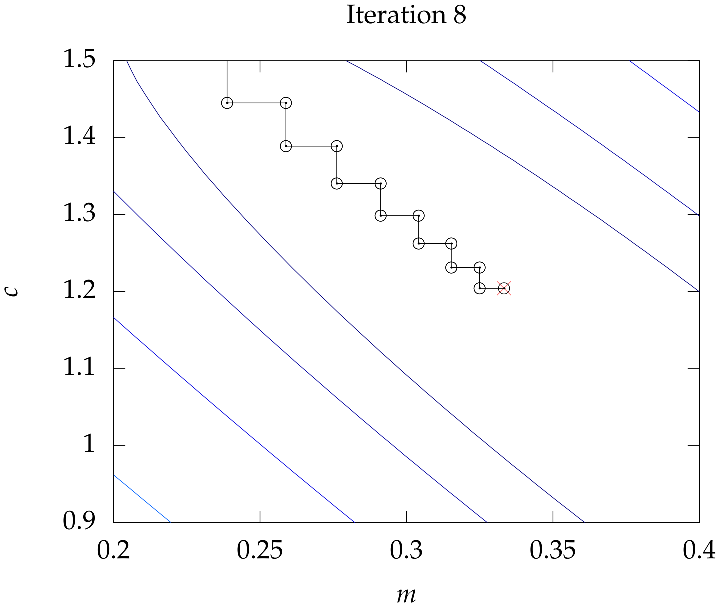
Coordinate Descent



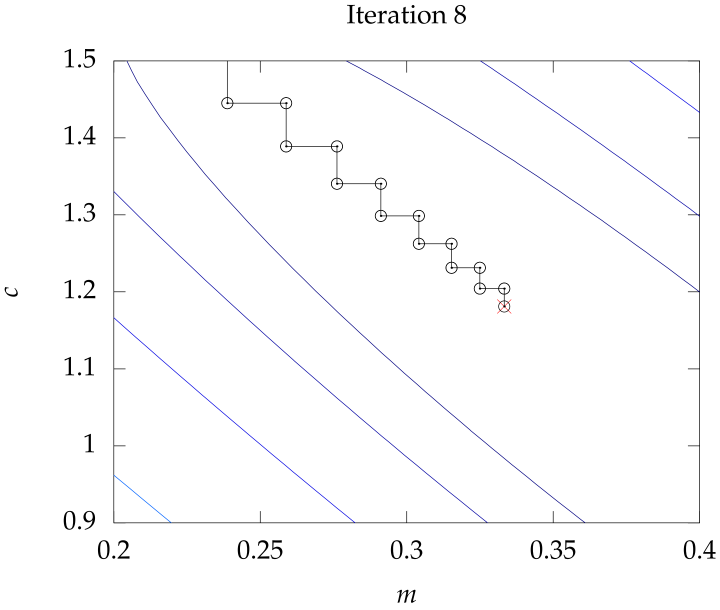
Coordinate Descent



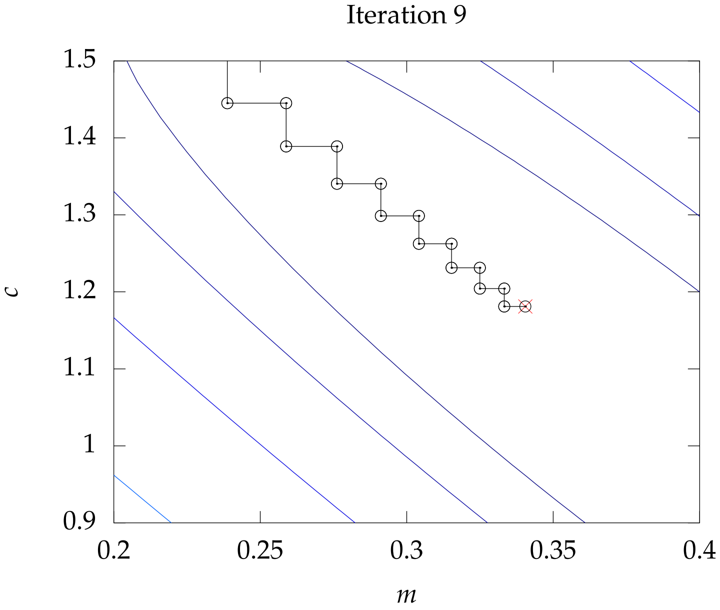
Coordinate Descent



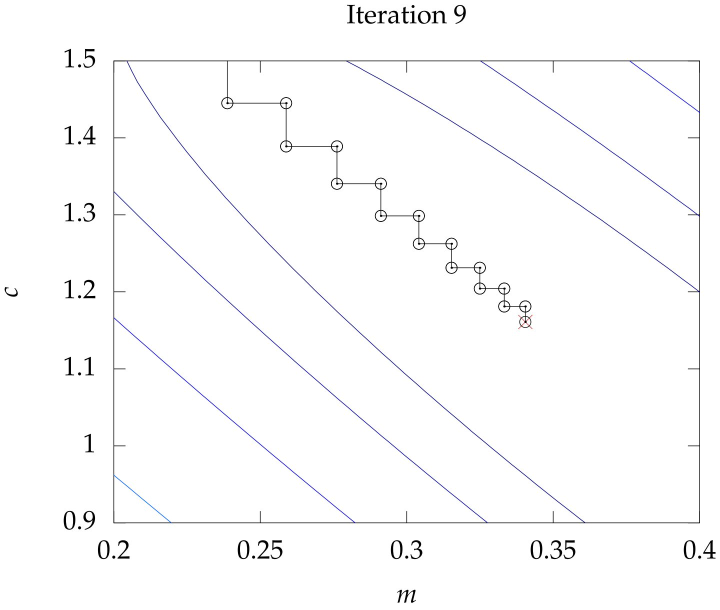
Coordinate Descent



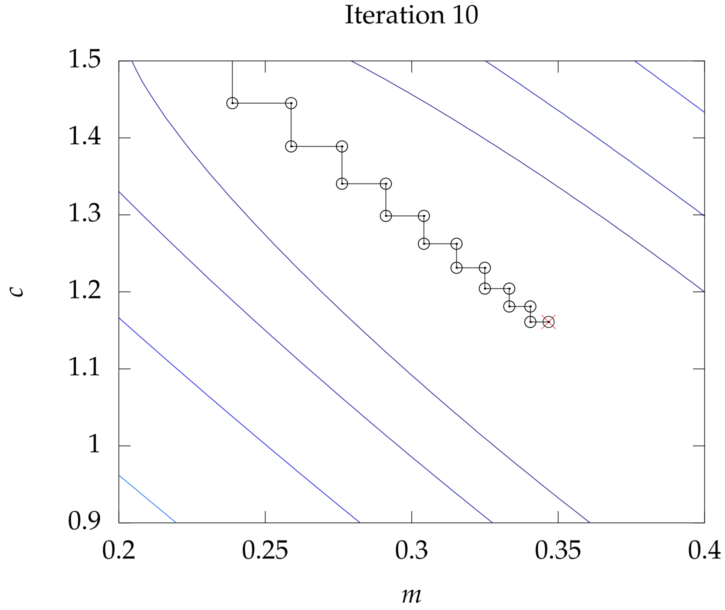
Coordinate Descent



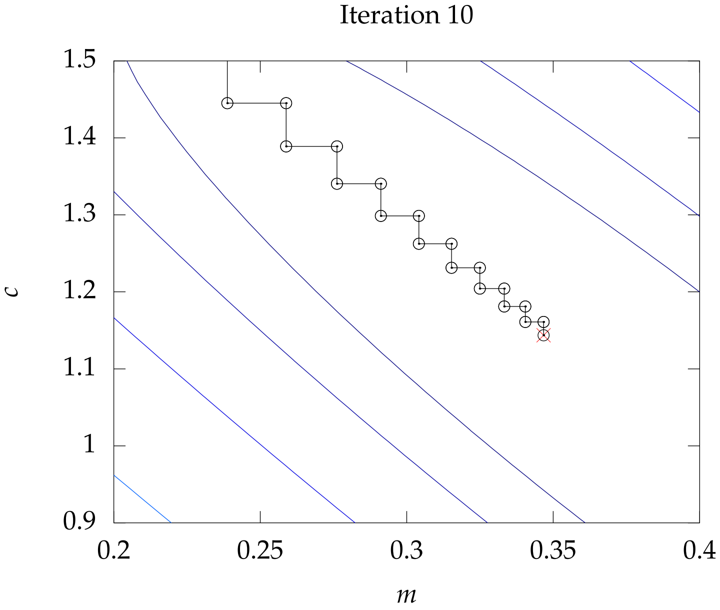
Coordinate Descent



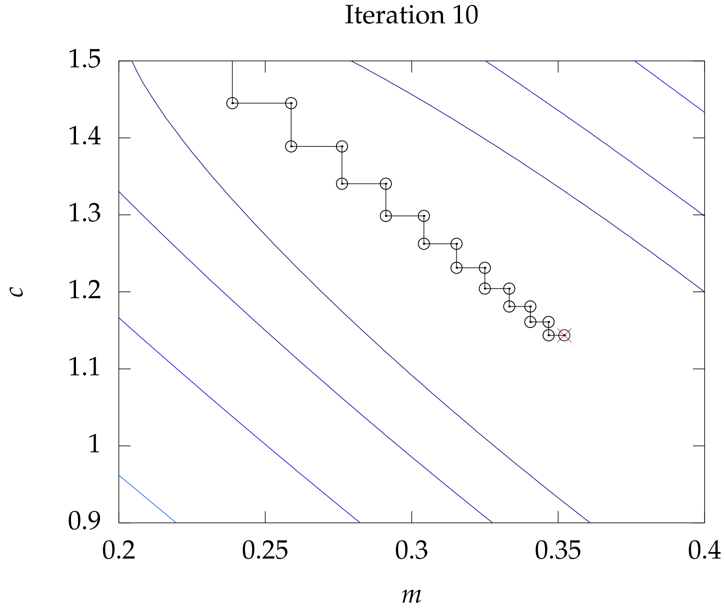
Coordinate Descent



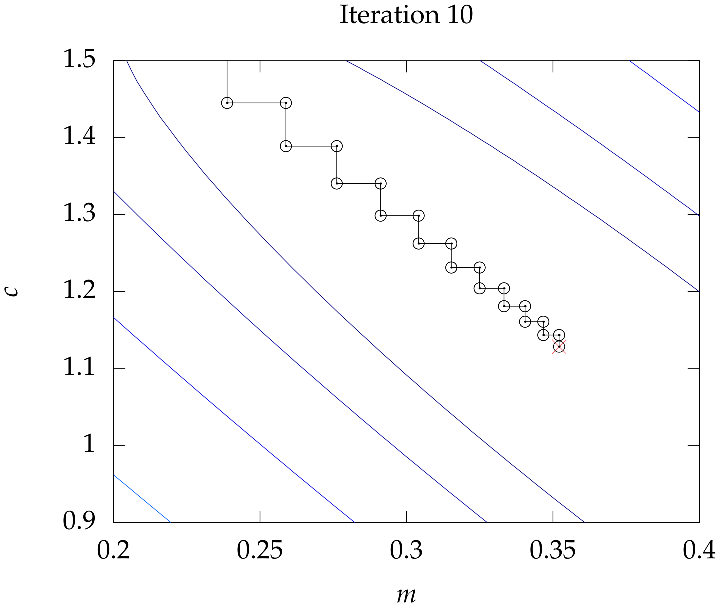
Coordinate Descent



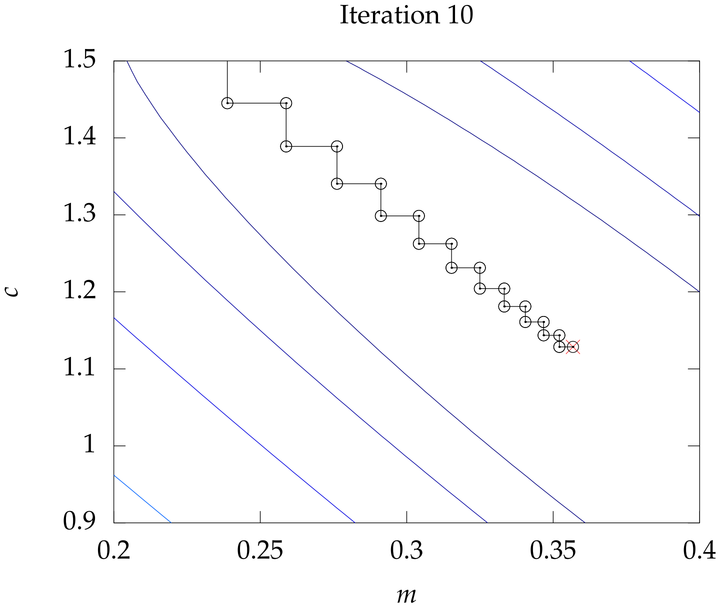
Coordinate Descent



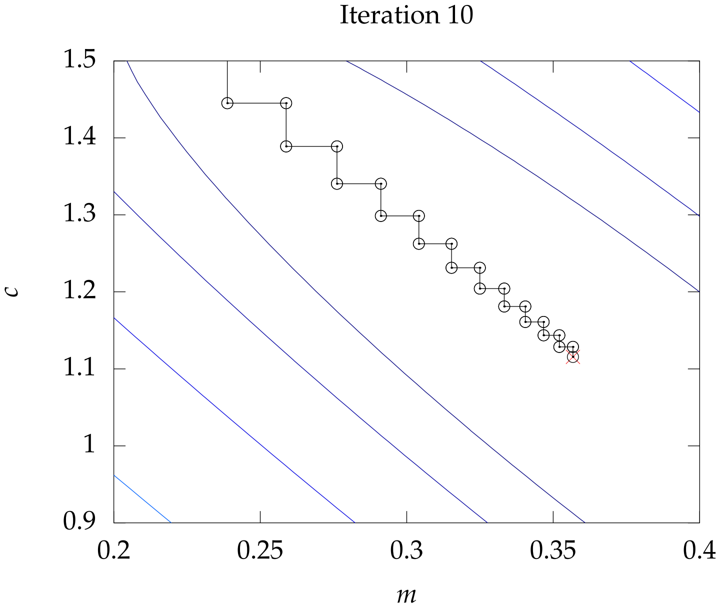
Coordinate Descent



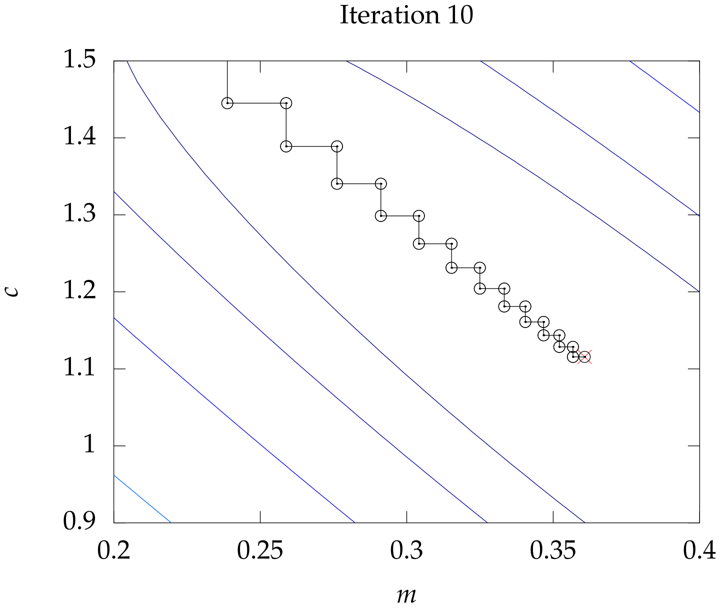
Coordinate Descent



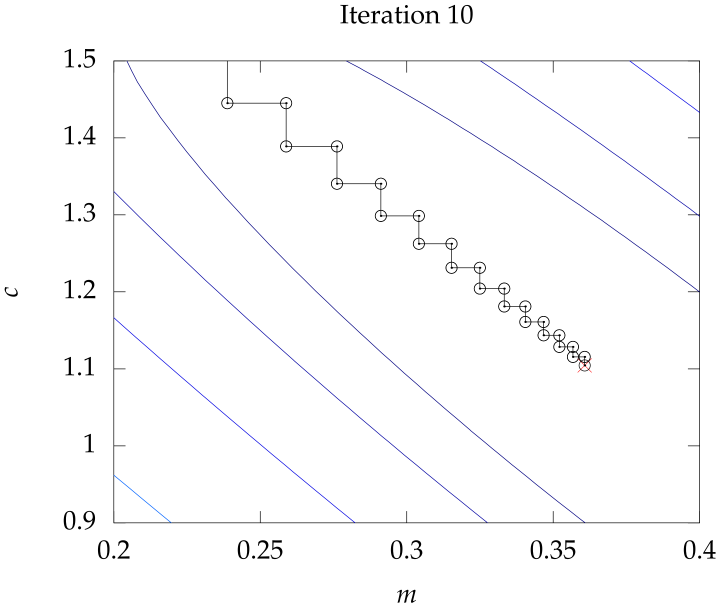
Coordinate Descent



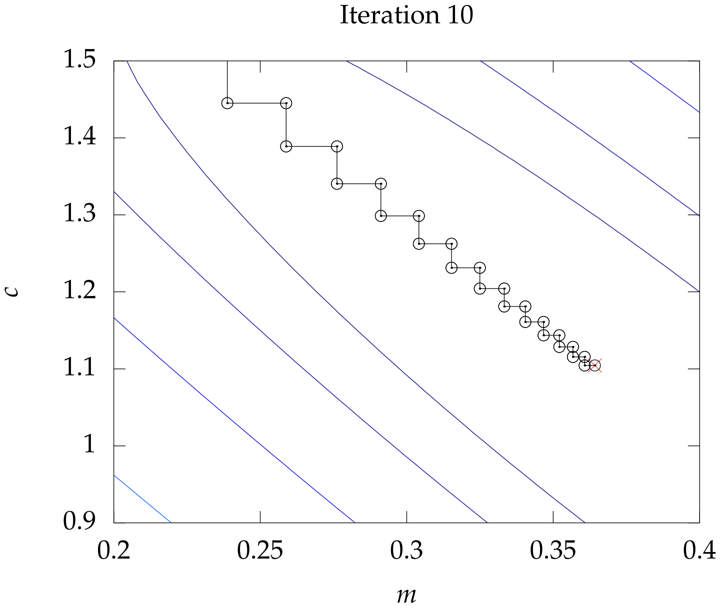
Coordinate Descent



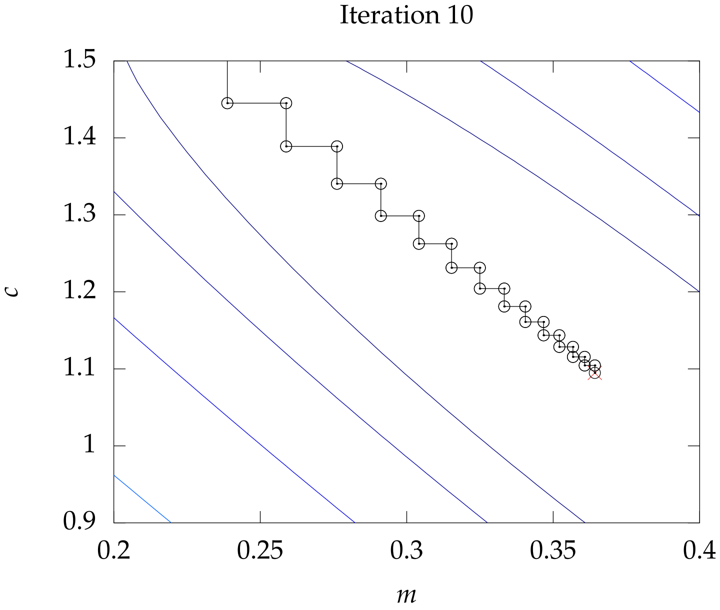
Coordinate Descent



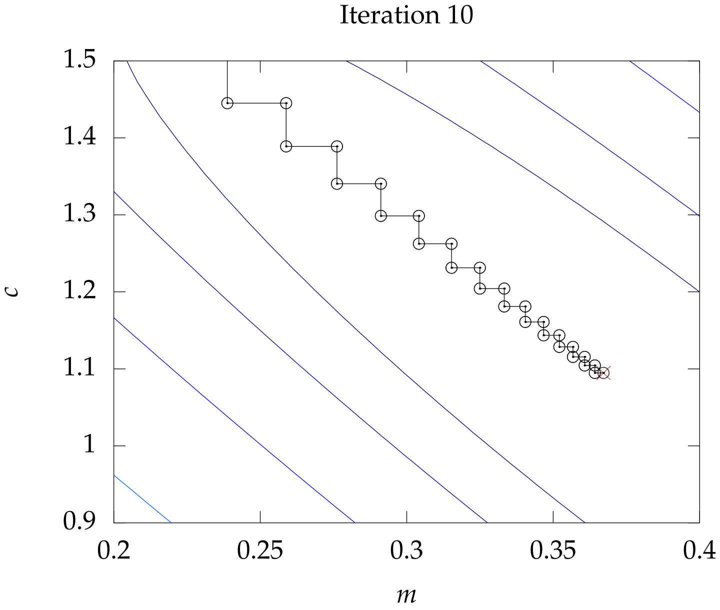
Coordinate Descent



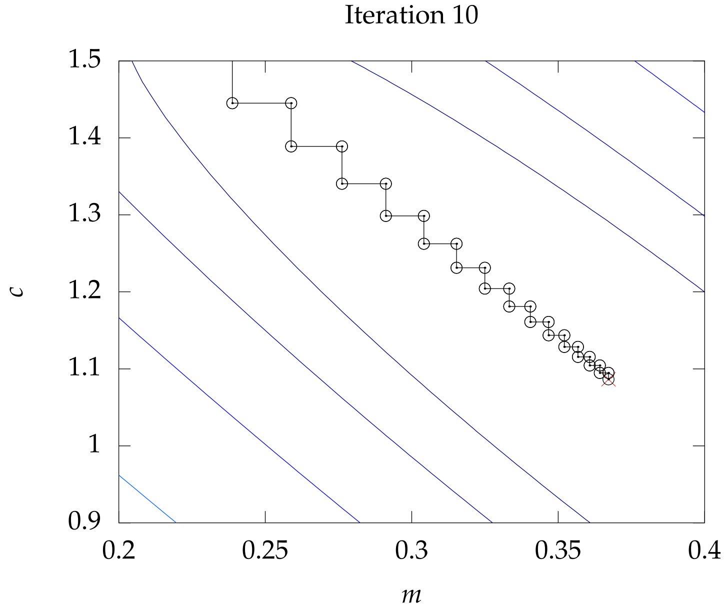
Coordinate Descent



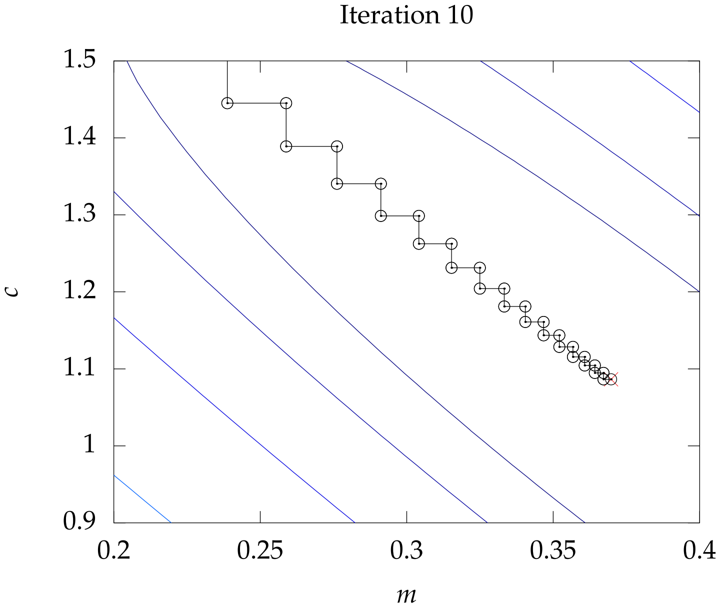
Coordinate Descent



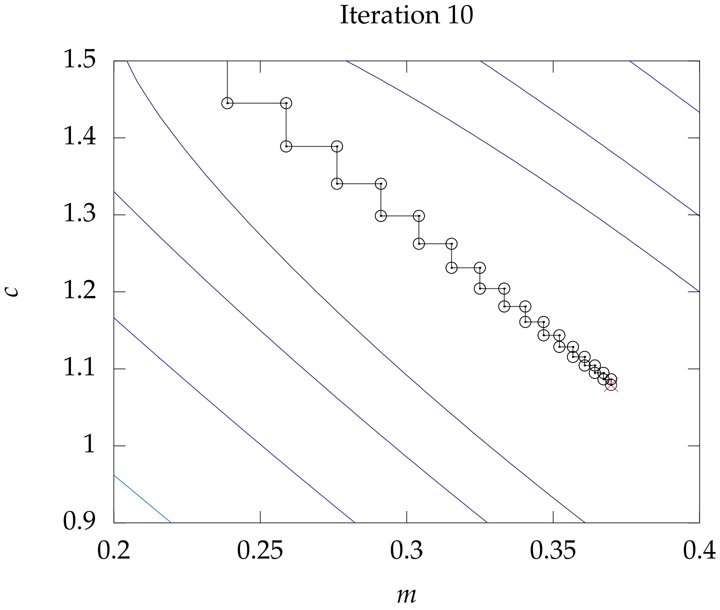
Coordinate Descent



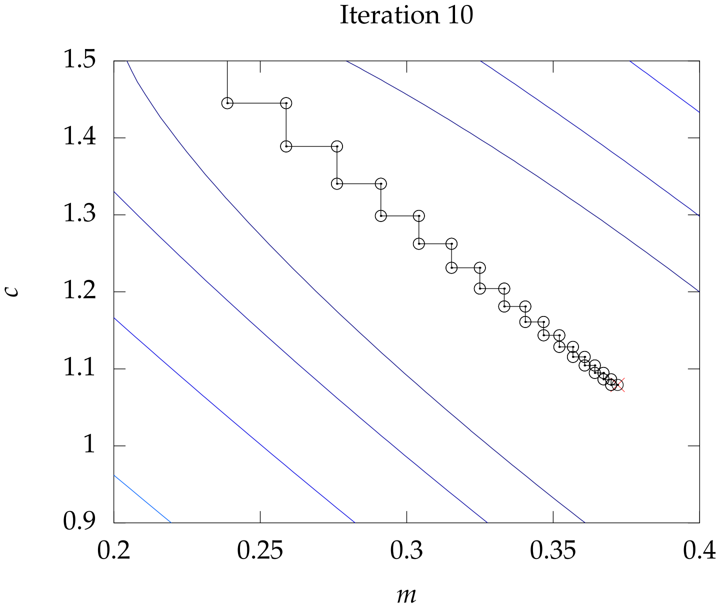
Coordinate Descent



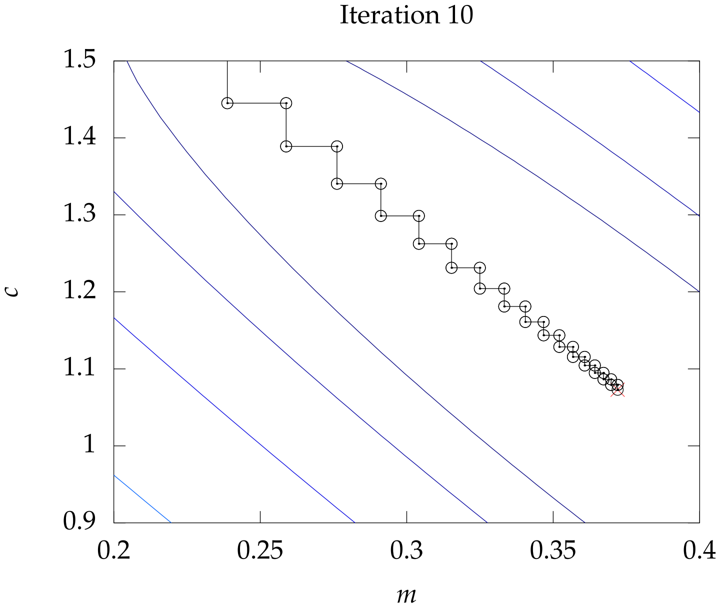
Coordinate Descent



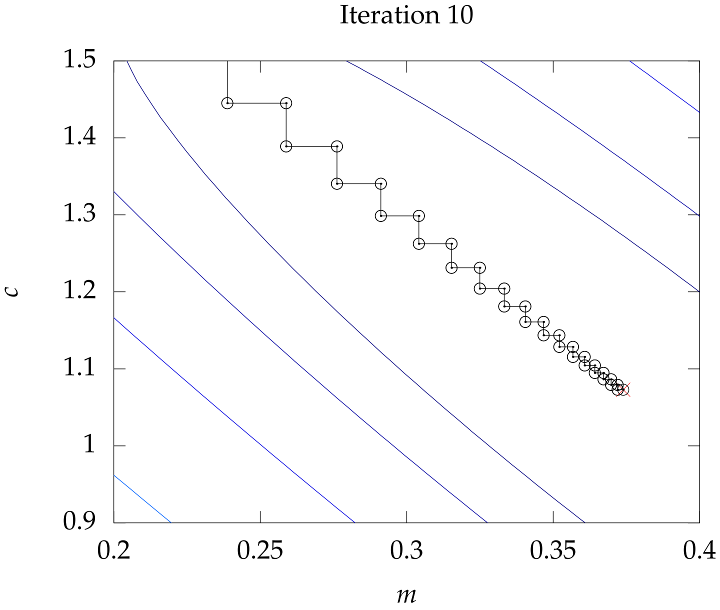
Coordinate Descent



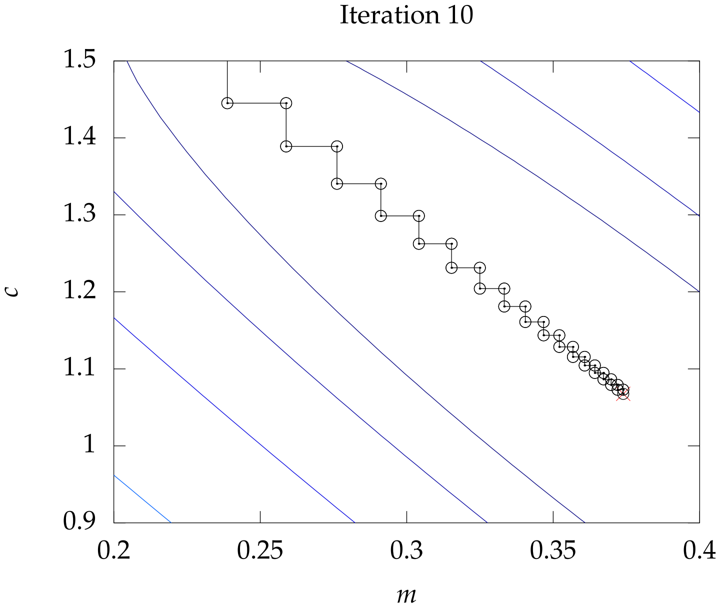
Coordinate Descent



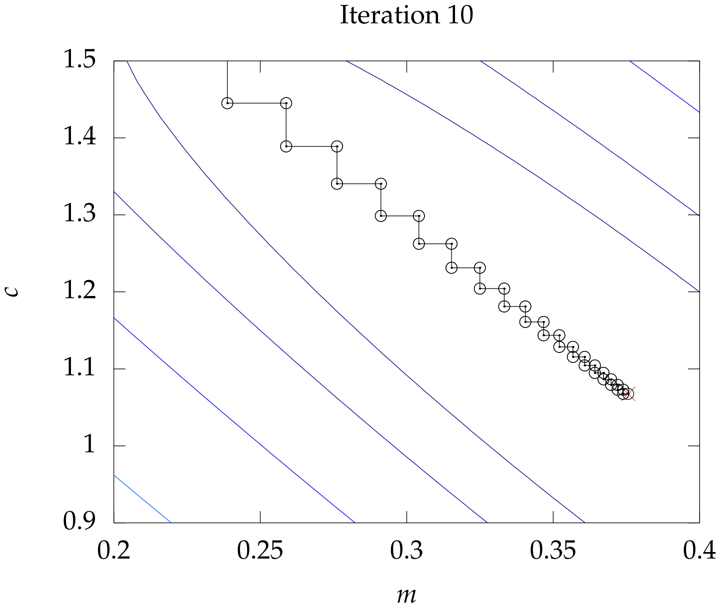
Coordinate Descent



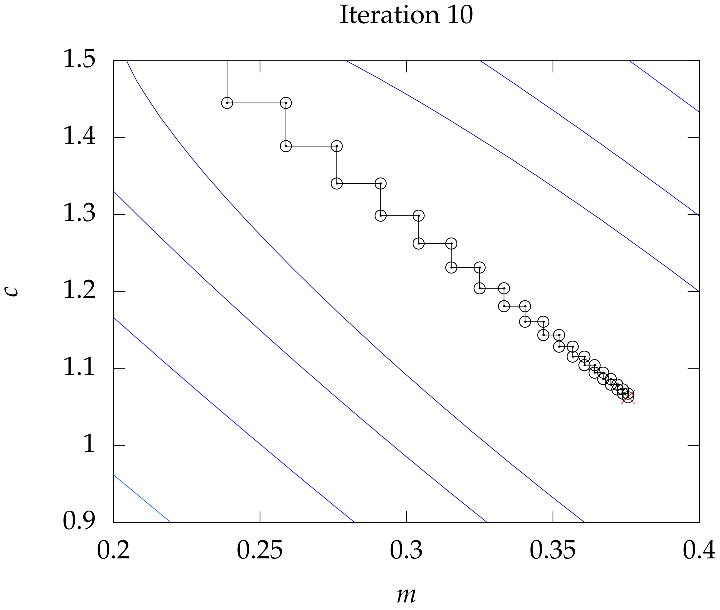
Coordinate Descent



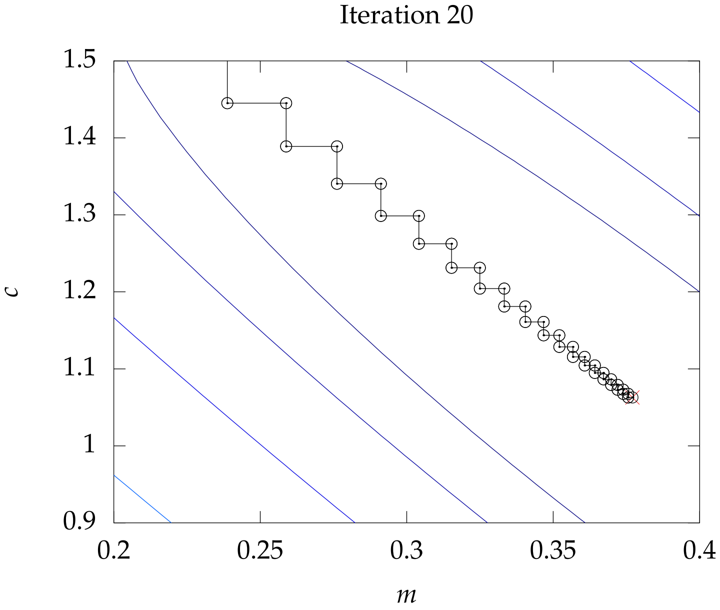
Coordinate Descent



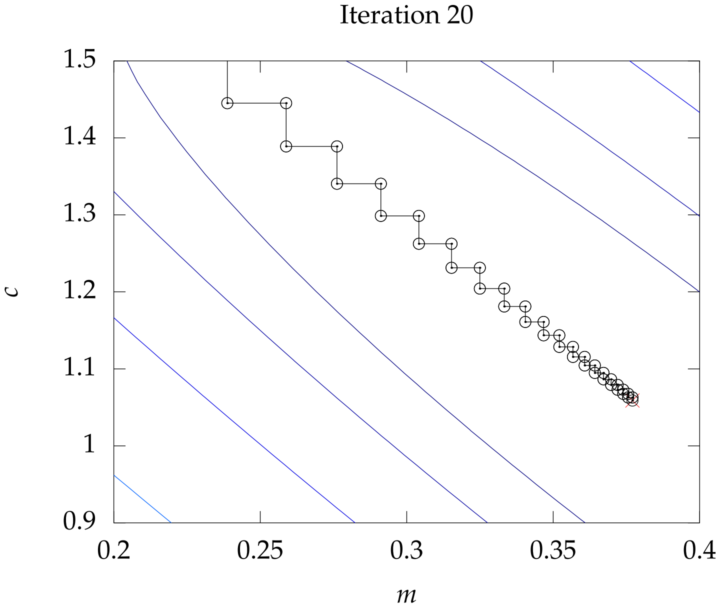
Coordinate Descent



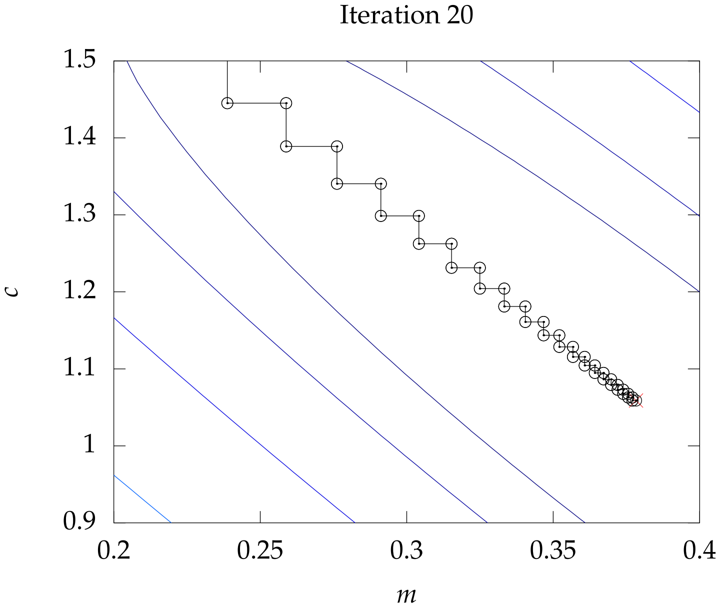
Coordinate Descent



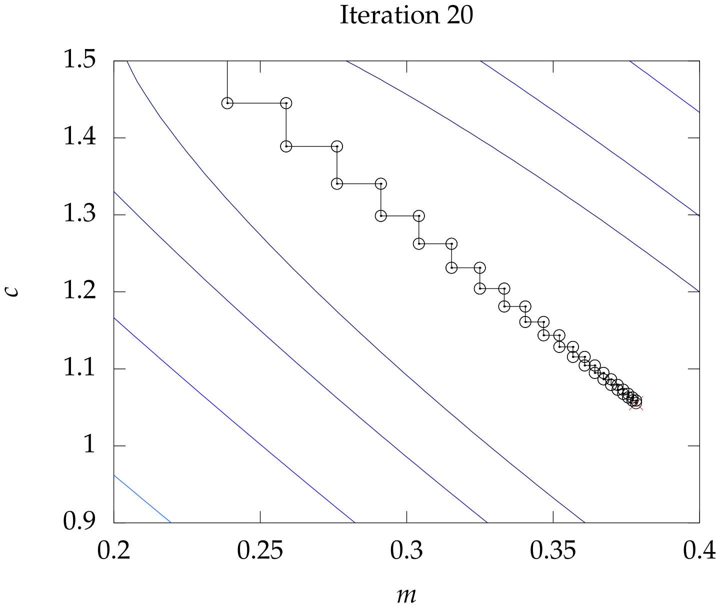
Coordinate Descent



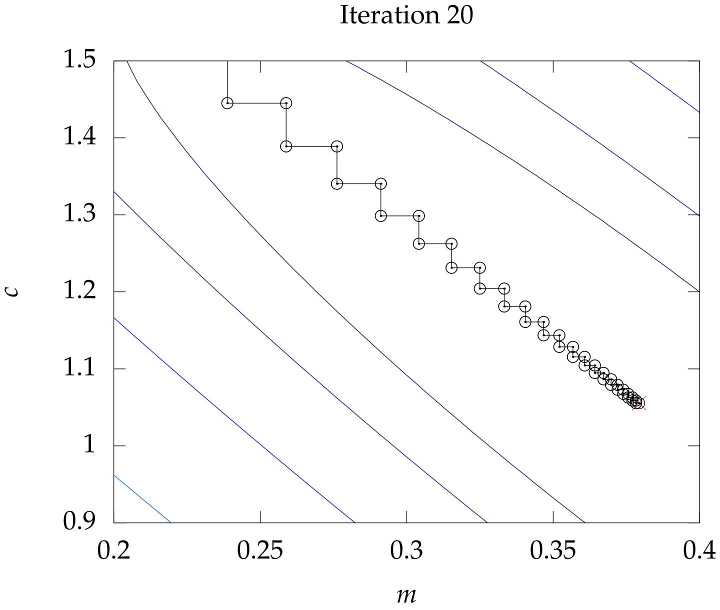
Coordinate Descent



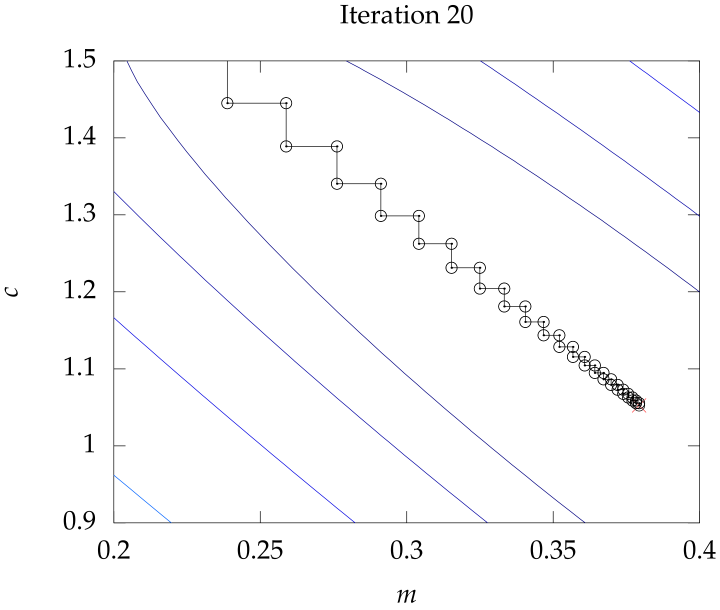
Coordinate Descent



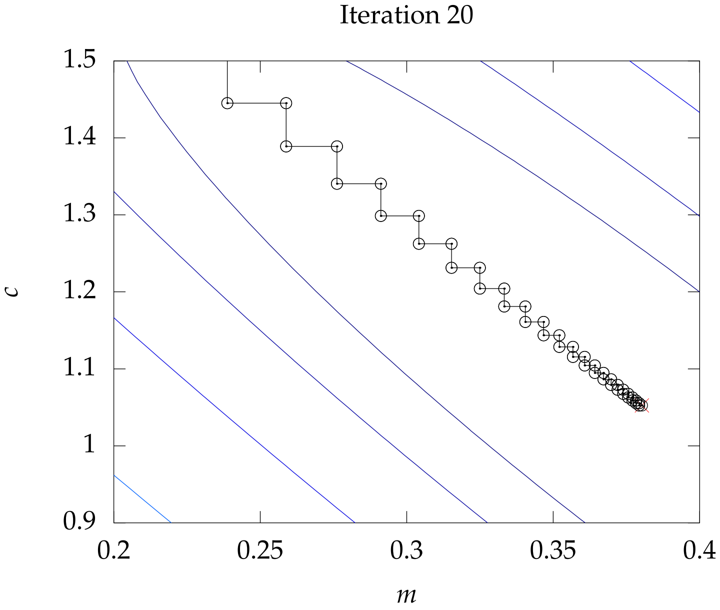
Coordinate Descent



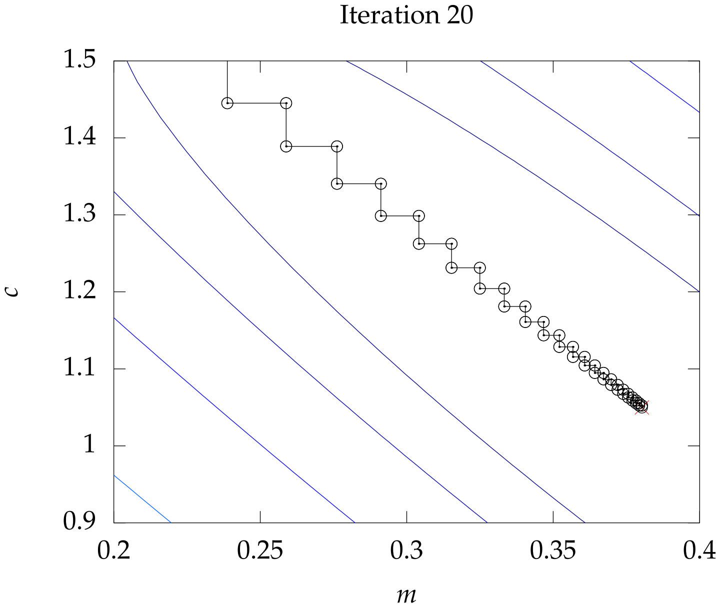
Coordinate Descent



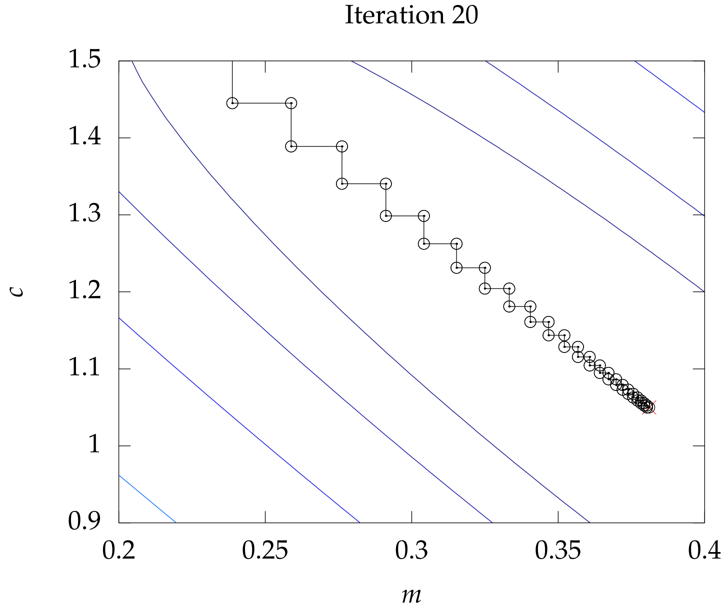
Coordinate Descent



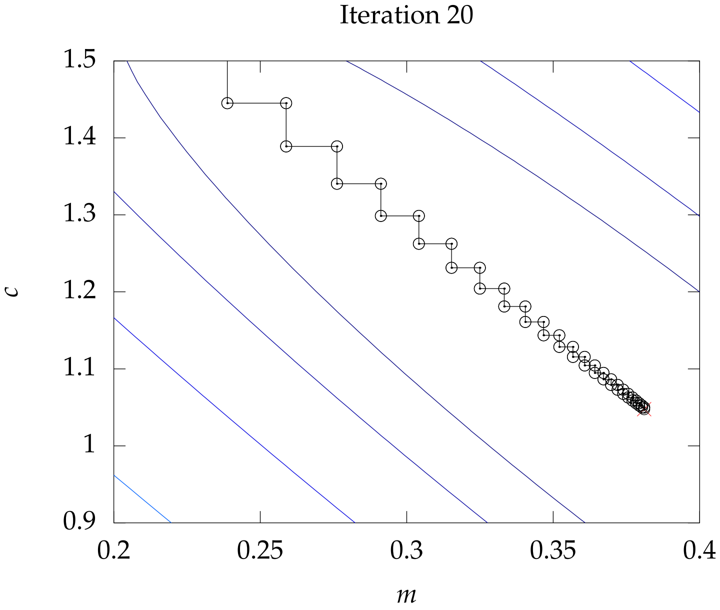
Coordinate Descent



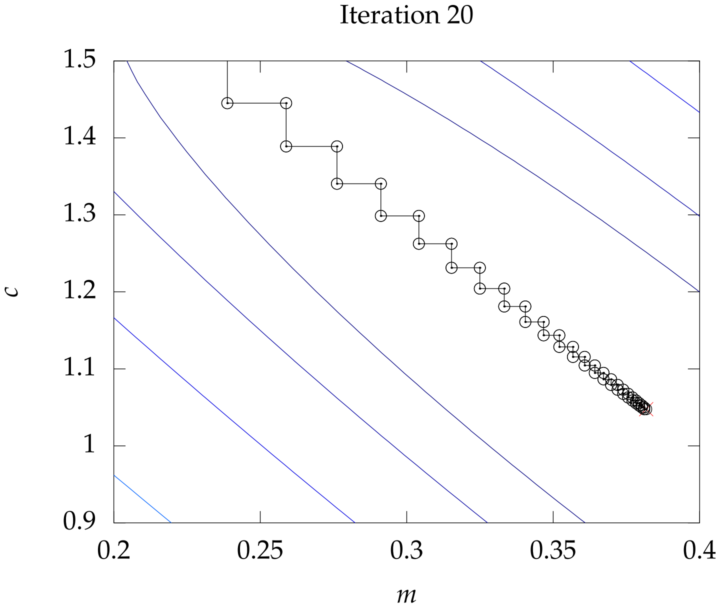
Coordinate Descent



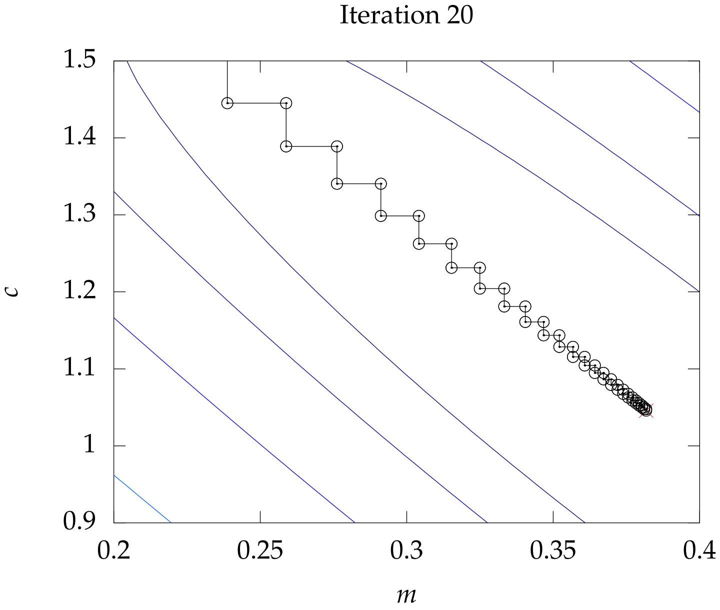
Coordinate Descent



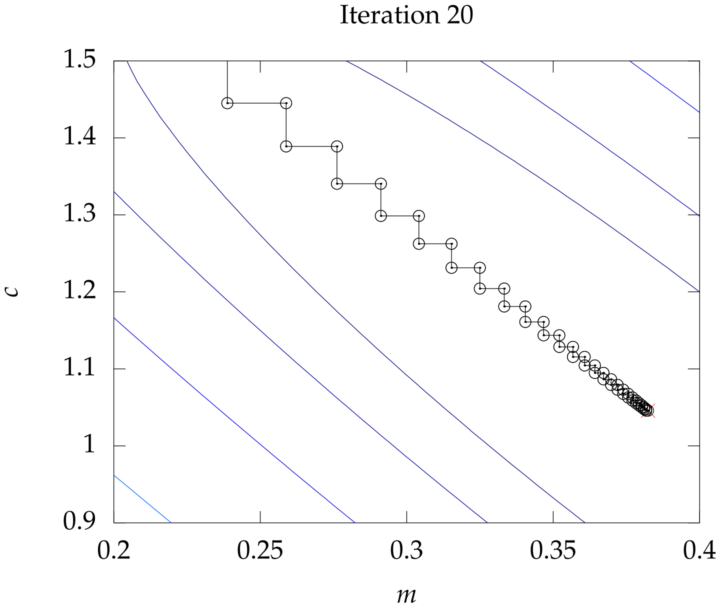
Coordinate Descent



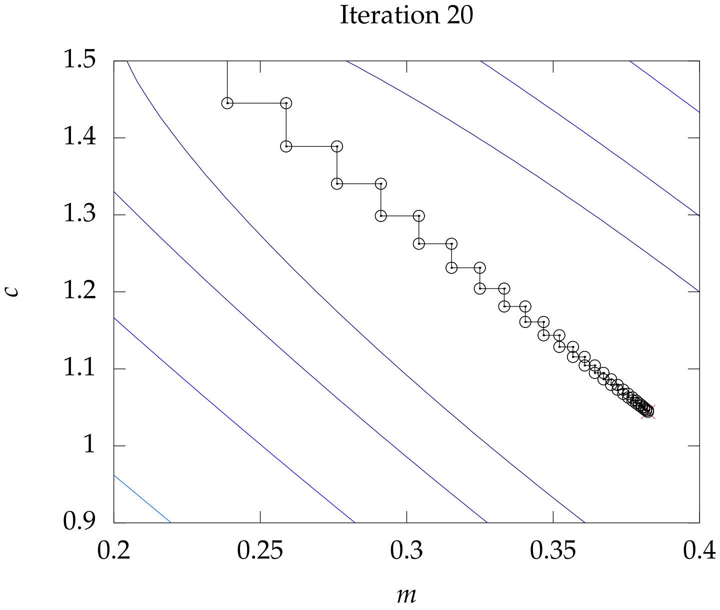
Coordinate Descent



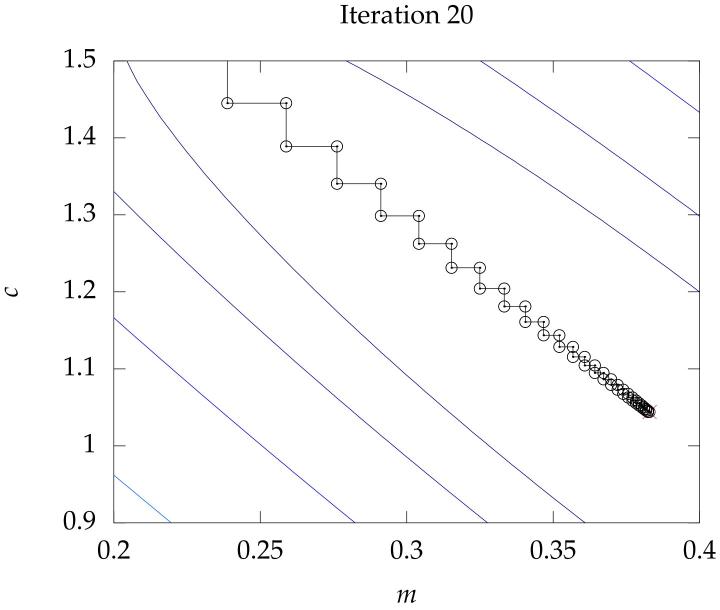
Coordinate Descent



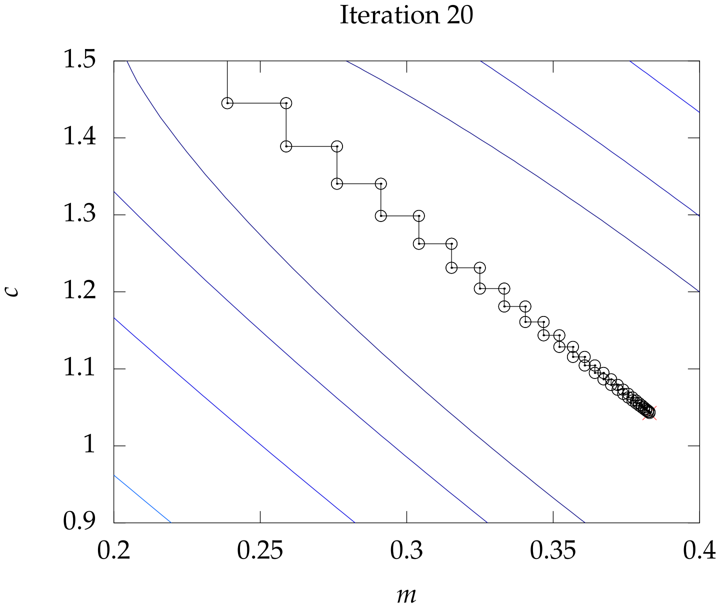
Coordinate Descent



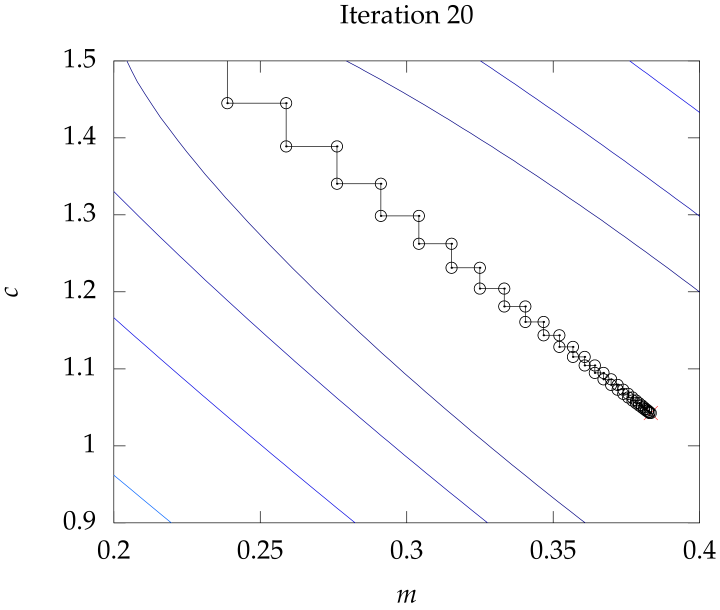
Coordinate Descent



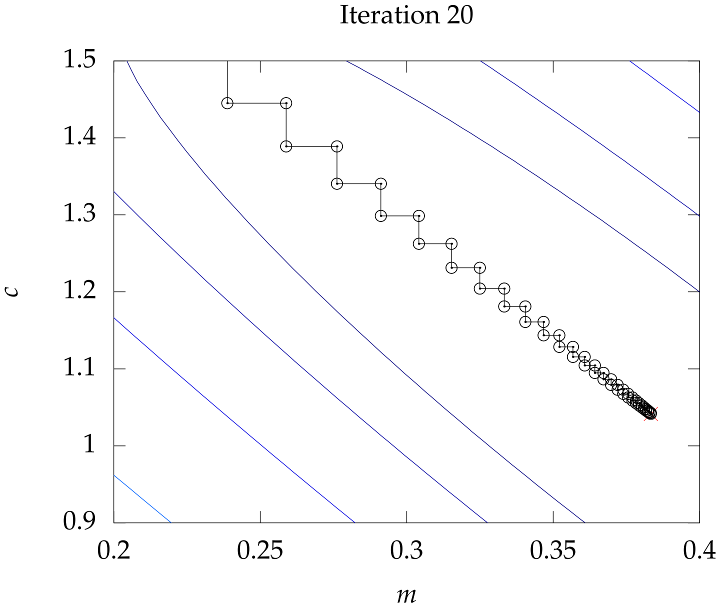
Coordinate Descent



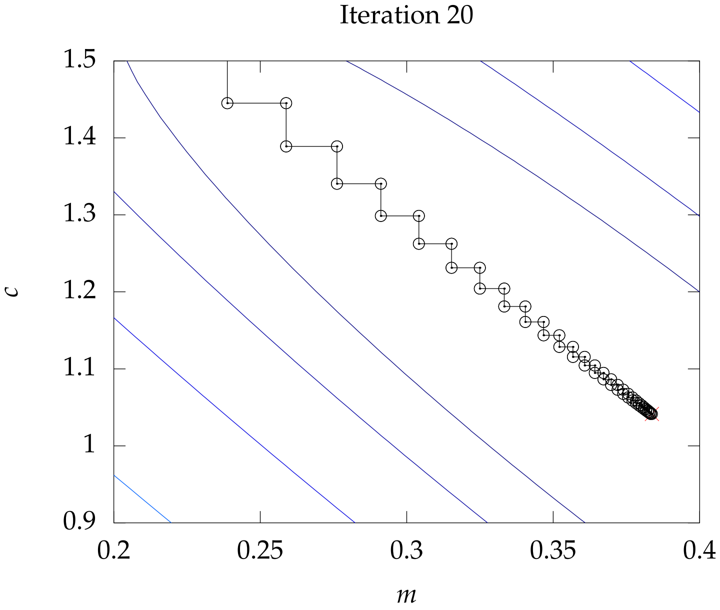
Coordinate Descent



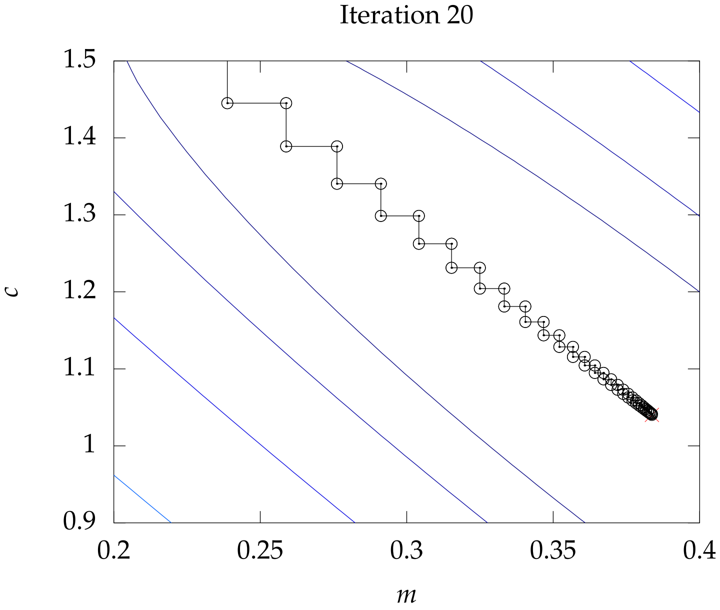
Coordinate Descent



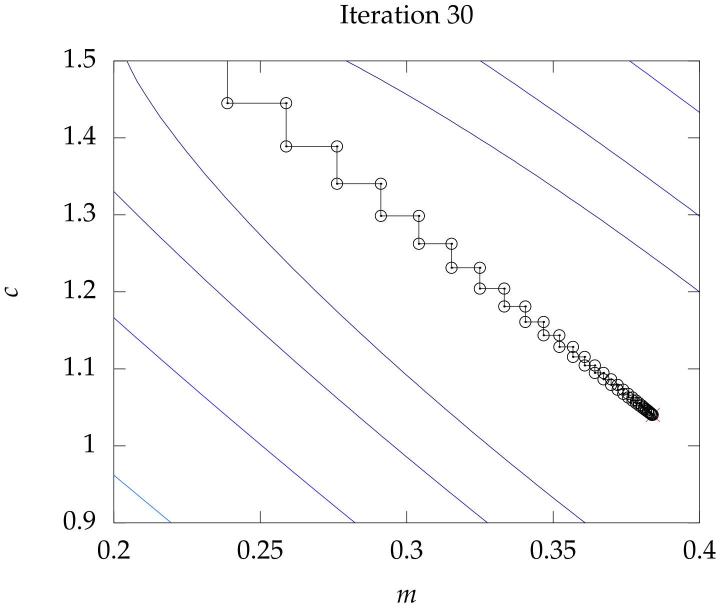
Coordinate Descent



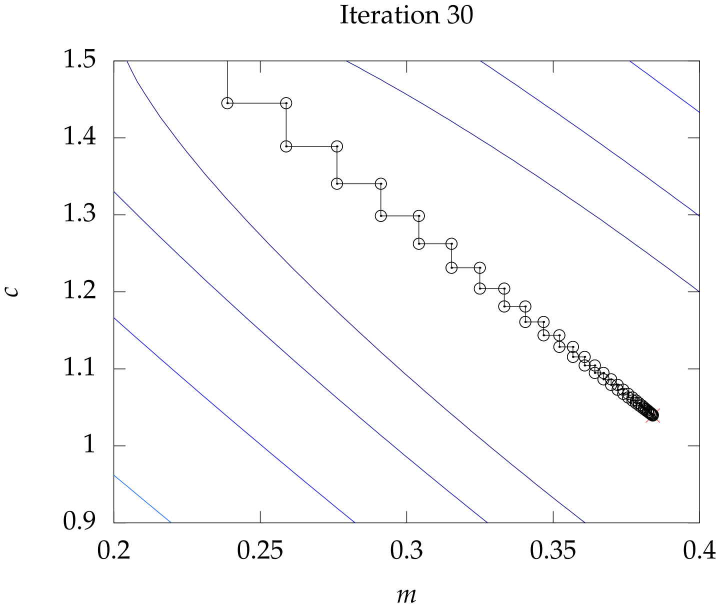
Coordinate Descent



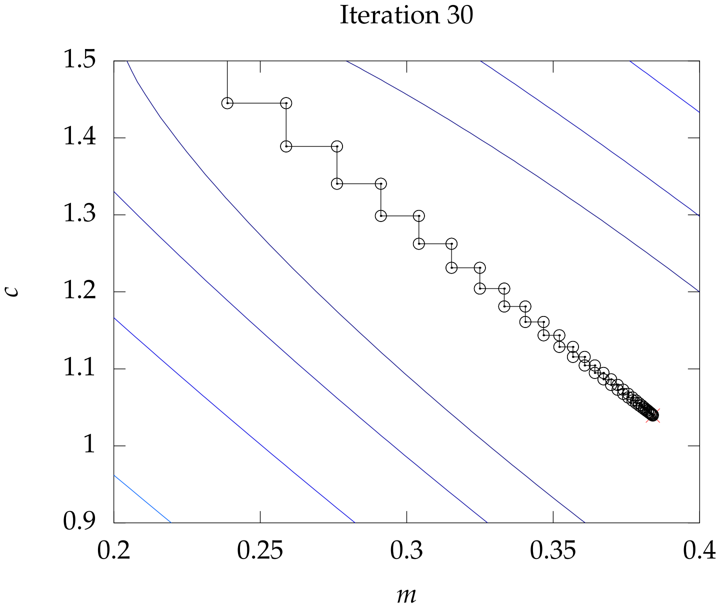
Coordinate Descent



Coordinate Descent



Coordinate Descent



Important Concepts Not Covered

- Optimization methods.
  - Second order methods, conjugate gradient, quasi-Newton and Newton.
  - Effective heuristics such as momentum, CMA, etc
- Local vs global solutions (Bayesian optimization!).

Learning is probabilistic modeling

Machine Learning and Probability

- The world is an *uncertain* place.

  Epistemic uncertainty: uncertainty arising through lack of knowledge. (What colour socks is that person wearing?)

  Aleatoric uncertainty: uncertainty arising through an underlying stochastic system. (Where will a sheet of paper fall if I drop it?)

Machine Learning and Probability

- The world is an *uncertain* place.

  Epistemic uncertainty: uncertainty arising through lack of knowledge. (What colour socks is that person wearing?)

  Aleatoric uncertainty: uncertainty arising through an underlying stochastic system. (Where will a sheet of paper fall if I drop it?)

Machine Learning and Probability

- The world is an *uncertain* place.

  Epistemic uncertainty: uncertainty arising through lack of knowledge. (What colour socks is that person wearing?)

  Aleatoric uncertainty: uncertainty arising through an underlying stochastic system. (Where will a sheet of paper fall if I drop it?)

Probability: A Framework to Characterise Uncertainty

- We need a framework to characterise the uncertainty.
- In this course we make use of probability theory to characterise uncertainty.

Probability: A Framework to Characterise Uncertainty

- We need a framework to characterise the uncertainty.
- In this course we make use of probability theory to characterise uncertainty.

Richard Price

- Welsh philosopher and essay writer.
- Edited **Thomas Bayes**'s essay which contained foundations of Bayesian philosophy.

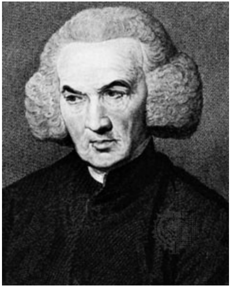


Figure: Richard Price, 1723–1791. (source Wikipedia)

Laplace

- French Mathematician and Astronomer.

Figure: Pierre-Simon Laplace, 1749–1827. (source Wikipedia)

Probabilistic Intepretation

- Quadratic error functions can be seen as Gaussian noise models [1, 2].
- Imagine we are seeing data given by,

$$y(x_i) = mx_i + c + \epsilon$$

where ϵ is Gaussian noise with standard deviation σ,

$$\epsilon \sim \mathcal{N}\left(0, \sigma^2\right).$$

Noise Corrupted Mapping

- This implies that

$$y_i \sim \mathcal{N}\left(mx_i + c, \sigma^2\right)$$

- Which we also write

$$p(y_i|\mathbf{w}, \sigma) = \mathcal{N}\left(y_i|mx_i + c, \sigma^2\right)$$

Gaussian Likelihood

- If the noise is sampled independently for each data point from the same density we have

$$p(\mathbf{y}|m, c, \sigma^2) = \prod_{i=1}^{n} \mathcal{N}\left(y_i|mx_i + c, \sigma^2\right)$$

- This is an i.i.d. assumption about the noise.
- Writing the functional form we have

$$p(\mathbf{y}|m, c, \sigma^2) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y_i - mx_i - c)^2}{2\sigma^2}\right)$$

Gaussian Likelihood

- If the noise is sampled independently for each data point from the same density we have

$$p(\mathbf{y}|m, c, \sigma^2) = \prod_{i=1}^{n} \mathcal{N}\left(y_i|mx_i + c, \sigma^2\right)$$

- This is an i.i.d. assumption about the noise.
- Writing the functional form we have

$$p(\mathbf{y}|m, c, \sigma^2) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y_i - mx_i - c)^2}{2\sigma^2}\right)$$

Gaussian Likelihood

- If the noise is sampled independently for each data point from the same density we have

$$p(\mathbf{y}|m, c, \sigma^2) = \prod_{i=1}^{n} \mathcal{N}\left(y_i|mx_i + c, \sigma^2\right)$$

- This is an i.i.d. assumption about the noise.
- Writing the functional form we have

$$p(\mathbf{y}|m, c, \sigma^2) \propto \prod_{i=1}^{n} \exp\left(-\frac{(y_i - mx_i - c)^2}{2\sigma^2}\right)$$

Gaussian Likelihood

- If the noise is sampled independently for each data point from the same density we have

$$p(\mathbf{y}|m, c, \sigma^2) = \prod_{i=1}^{n} \mathcal{N}\left(y_i|mx_i + c, \sigma^2\right)$$

- This is an i.i.d. assumption about the noise.
- Writing the functional form we have

$$p(\mathbf{y}|m, c, \sigma^2) \propto \prod_{i=1}^{n} \exp\left(-\frac{(y_i - mx_i - c)^2}{2\sigma^2}\right)$$

Gaussian Likelihood

- If the noise is sampled independently for each data point from the same density we have

$$p(\mathbf{y}|m, c, \sigma^2) = \prod_{i=1}^{n} \mathcal{N}\left(y_i|mx_i + c, \sigma^2\right)$$

- This is an i.i.d. assumption about the noise.
- Writing the functional form we have

$$p(\mathbf{y}|m, c, \sigma^2) \propto \exp\left(-\sum_{i=1}^{n} \frac{(y_i - mx_i - c)^2}{2\sigma^2}\right)$$

Gaussian Likelihood

- If the noise is sampled independently for each data point from the same density we have

$$p(\mathbf{y}|m, c, \sigma^2) = \prod_{i=1}^{n} \mathcal{N}\left(y_i|mx_i + c, \sigma^2\right)$$

- This is an i.i.d. assumption about the noise.
- Writing the functional form we have

$$p(\mathbf{y}|m, c, \sigma^2) \propto \exp\left(-\sum_{i=1}^{n} \frac{(y_i - mx_i - c)^2}{2\sigma^2}\right)$$

Gaussian Likelihood

- If the noise is sampled independently for each data point from the same density we have

$$p(\mathbf{y}|m, c, \sigma^2) = \prod_{i=1}^{n} \mathcal{N}\left(y_i|mx_i + c, \sigma^2\right)$$

- This is an i.i.d. assumption about the noise.
- Writing the functional form we have

$$p(\mathbf{y}|m, c, \sigma^2) \propto \exp\left(-\sum_{i=1}^{n} \frac{(y_i - mx_i - c)^2}{2\sigma^2}\right)$$

Gaussian Log Likelihood

- If the noise is sampled independently for each data point from the same density we have

$$p(\mathbf{y}|m, c, \sigma^2) = \prod_{i=1}^{n} \mathcal{N}\left(y_i|mx_i + c, \sigma^2\right)$$

- This is an i.i.d. assumption about the noise.
- Writing the functional form we have

$$\log p(\mathbf{y}|m, c, \sigma^2) = -\frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i - mx_i - c)^2 + \text{const}$$

Gaussian Log Likelihood

- If the noise is sampled independently for each data point from the same density we have

$$p(\mathbf{y}|m, c, \sigma^2) = \prod_{i=1}^{n} \mathcal{N}\left(y_i|mx_i + c, \sigma^2\right)$$

- This is an i.i.d. assumption about the noise.
- Writing the functional form we have

$$-\log p(\mathbf{y}|m, c, \sigma^2) = \frac{1}{2\sigma^2} \sum_{i=1}^{n} (y_i - mx_i - c)^2 + \text{const}$$

Gaussian Log Likelihood

- If the noise is sampled independently for each data point from the same density we have

$$p(\mathbf{y}|m, c, \sigma^2) = \prod_{i=1}^{n} \mathcal{N}\left(y_i|mx_i + c, \sigma^2\right)$$

- This is an i.i.d. assumption about the noise.
- Writing the functional form we have

$$-\log p(\mathbf{y}|m, c, \sigma^2) = \frac{1}{2\sigma^2} E(m, c) + \text{const}$$

Probabilistic Interpretation of the Error Function

- Probabilistic Interpretation for Error Function is Negative Log Likelihood.
- *Minimizing* error function is equivalent to *maximizing* log likelihood.
- Maximizing *log likelihood* is equivalent to maximizing the *likelihood* because log is monotonic.
- Probabilistic interpretation: Minimizing error function is equivalent to maximum likelihood with respect to parameters.

Sample Based Approximation implies i.i.d

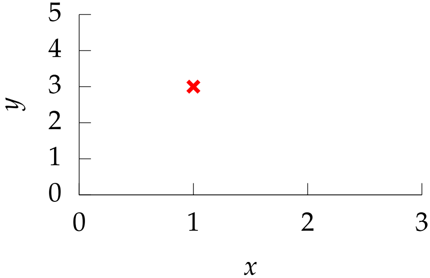
- The log likelihood is
$$L(\boldsymbol{\theta}) = \log P(\mathbf{y}|\boldsymbol{\theta})$$
- If the likelihood is *independent* over the individual data points,
$$P(\mathbf{y}|\boldsymbol{\theta}) = \prod_{i=1}^{n} P(y_i|\boldsymbol{\theta})$$
- This is equivalent to the assumption that the data is *independent* and *identically* distributed. This is known as *i.i.d.*.
- Now the log likelihood is
$$L(\boldsymbol{\theta}) = \sum_{i=1}^{n} \log P(y_i|\boldsymbol{\theta})$$
- We take the negative log likelihood to recover the sum of squares error.

Bayesian perspective

Underdetermined System

What about two unknowns and *one* observation?

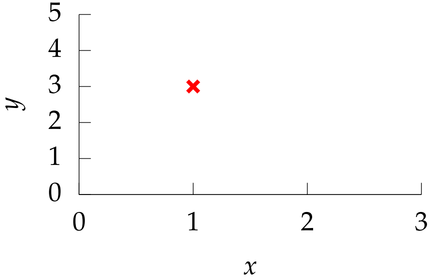
$$y_1 = mx_1 + c$$



Underdetermined System

Can compute m given c.

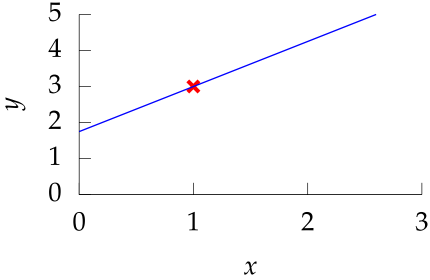
$$m = \frac{y_1 - c}{x}$$



Underdetermined System

Can compute m given c.

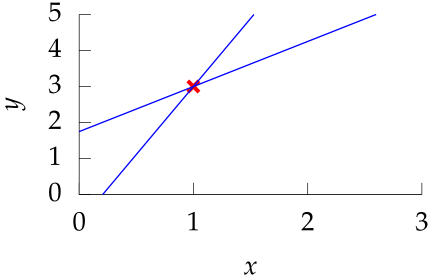
$$c = 1.75 \implies m = 1.25$$



Underdetermined System

Can compute m given c.

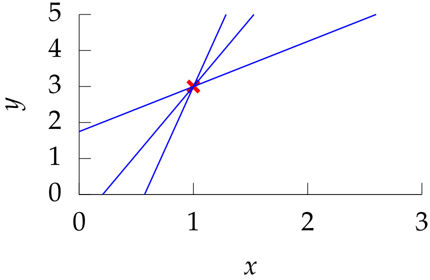
$$c = -0.777 \implies m = 3.78$$



Underdetermined System

Can compute m given c.

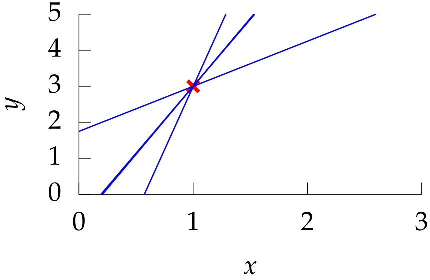
$$c = -4.01 \implies m = 7.01$$



Underdetermined System

Can compute m given c.

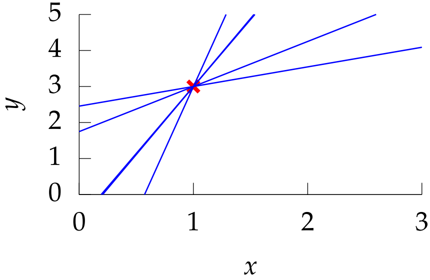
$$c = -0.718 \implies m = 3.72$$



Underdetermined System

Can compute m given c.

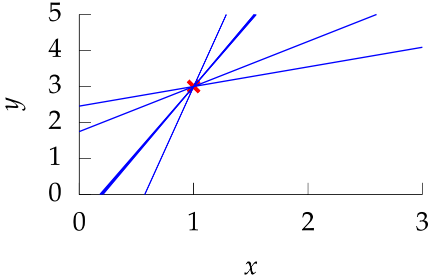
$$c = 2.45 \implies m = 0.545$$



Underdetermined System

Can compute m given c.

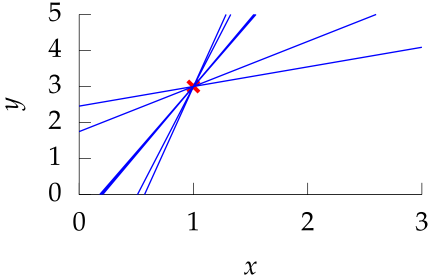
$$c = -0.657 \implies m = 3.66$$



Underdetermined System

Can compute m given c.

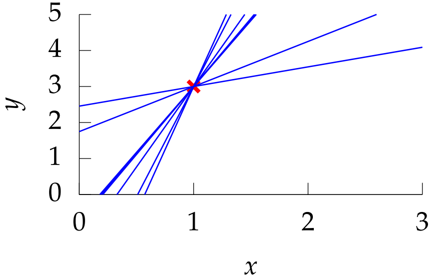
$$c = -3.13 \implies m = 6.13$$



Underdetermined System

Can compute m given c.

$$c = -1.47 \Longrightarrow m = 4.47$$

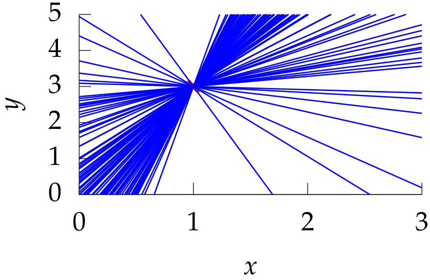


Underdetermined System

Can compute m given c. Assume

$$c \sim \mathcal{N}(0, 4),$$

we find a distribution of solutions.



Bayesian Approach

- Likelihood for the regression example has the form

$$p(\mathbf{y}|\mathbf{w}, \sigma^2) = \prod_{i=1}^{n} \mathcal{N}\left(y_i|\mathbf{w}^\top \boldsymbol{\phi}_i, \sigma^2\right).$$

- Suggestion was to maximize this likelihood with respect to **w**.
- This can be done with gradient based optimization of the log likelihood.
- Alternative approach: integration across **w**.
- Consider expected value of likelihood under a range of potential **w**s.
- This is known as the *Bayesian* approach.

Note on the Term Bayesian

- We will use Bayes' rule to invert probabilities in the Bayesian approach.
  - Bayesian is not named after Bayes' rule (v. common confusion).
  - The term Bayesian refers to the treatment of the parameters as stochastic variables.
  - This approach was proposed by Laplace and Bayes independently.
  - For early statisticians this was very controversial (Fisher et al).

Bayesian Controversy

- Bayesian controversy relates to treating *epistemic* uncertainty as *aleatoric* uncertainty.
- Another analogy:
  - Before a football match the uncertainty about the result is *aleatoric*.
  - If I watch a recorded match *without* knowing the result the uncertainty is *epistemic*.

Simple Bayesian Inference

$$\text{posterior} = \frac{\text{likelihood} \times \text{prior}}{\text{marginal likelihood}}$$

- Four components:
  1. Prior distribution: represents belief about parameter values before seeing data.
  2. Likelihood: gives relation between parameters and data.
  3. Posterior distribution: represents updated belief about parameters after data is observed.
  4. Marginal likelihood: represents assessment of the quality of the model. Can be compared with other models (likelihood/prior combinations). Ratios of marginal likelihoods are known as Bayes factors.

Recap

- We can see the learning process as an optimization problem.
- We can also see the learning process as probabilistic modelling (that also depends on parameters that need to be optimised).
- The Bayesian frameworks allows us to handle ‘epistemic’ uncertainty in systems.
- Examples only for linear functions, so far.

Generalizations: What if we want to use a more expressive model (non-linear function)

- ML as optimization: Regularization in RKHSs (kernel methods)
- Bayesian perspective: Gaussian Processes.

Both approaches are related: as standard regression and Bayesian regression are.

More important for us: Gaussian processes.

- Problem with Linear Regression—$\mathbf{x}$ may not be linearly related to $\mathbf{y}$.
- Potential solution: create a feature space: define $\phi(\mathbf{x})$ where $\phi(\cdot)$ is a nonlinear function of $\mathbf{x}$.
- Model for target is a linear combination of these nonlinear functions

$$f(\mathbf{x}) = \sum_{j=1}^{K} w_j \phi_j(\mathbf{x}) \tag{1}$$

Quadratic Basis

- Basis functions can be global. E.g. quadratic basis:

$$[1, x, x^2]$$

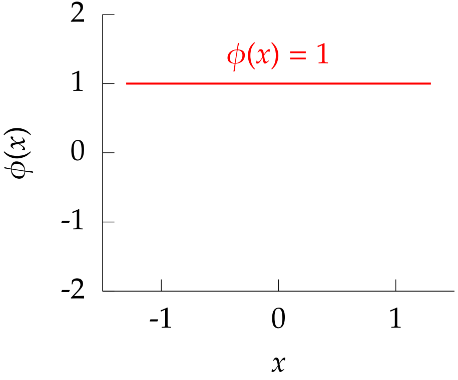


Figure: A quadratic basis.

Quadratic Basis

- Basis functions can be global. E.g. quadratic basis:

$$[1, x, x^2]$$

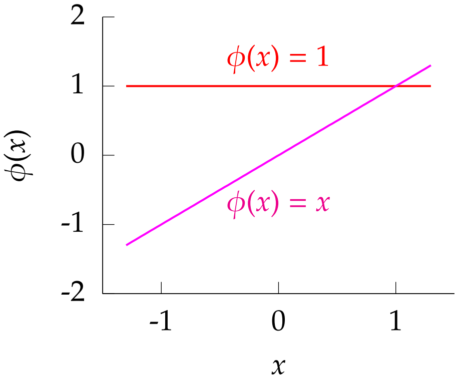


Figure: A quadratic basis.

Quadratic Basis

- Basis functions can be global. E.g. quadratic basis:

$$[1, x, x^2]$$

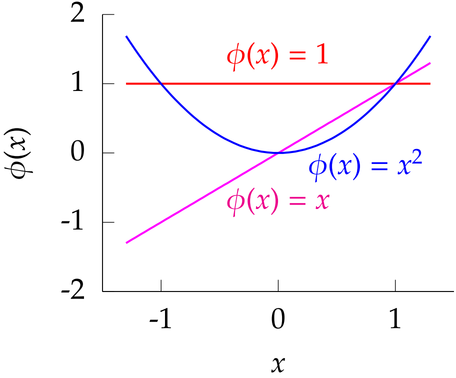


Figure: A quadratic basis.

Functions Derived from Quadratic Basis

$$f(x) = w_1 + w_2 x + w_3 x^2$$

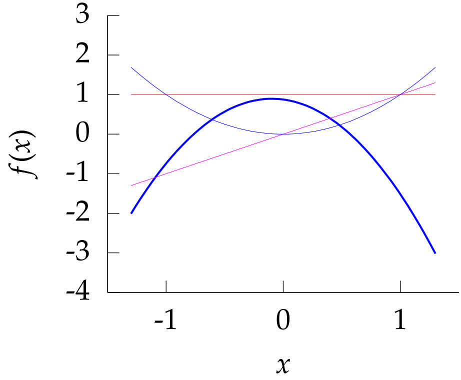


Figure: Function from quadratic basis with weights $w_1 = 0.87466$, $w_2 = -0.38835$, $w_3 = -2.0058$.

Functions Derived from Quadratic Basis

$$f(x) = w_1 + w_2 x + w_3 x^2$$

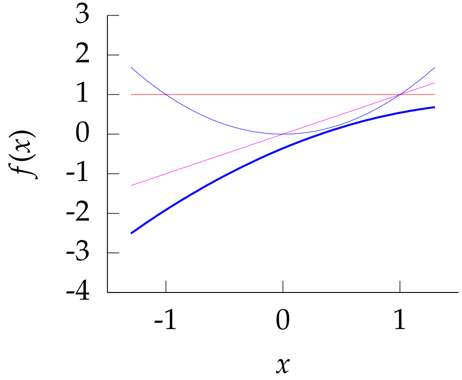


Figure: Function from quadratic basis with weights $w_1 = -0.35908$, $w_2 = 1.2274$, $w_3 = -0.32825$.

Functions Derived from Quadratic Basis

$$f(x) = w_1 + w_2 x + w_3 x^2$$

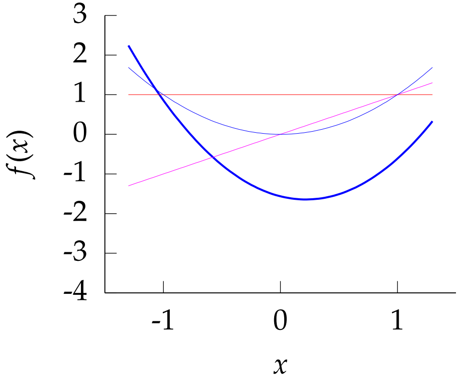


Figure: Function from quadratic basis with weights $w_1 = -1.5638$, $w_2 = -0.73577$, $w_3 = 1.6861$.

Radial Basis Functions

- Or they can be local. E.g. radial (or Gaussian) basis $\phi_j(x) = \exp\left(-\frac{(x-\mu_j)^2}{\ell^2}\right)$

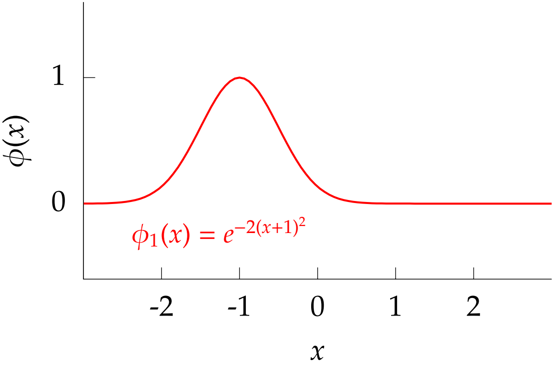


Figure: Radial basis functions.

Radial Basis Functions

- Or they can be local. E.g. radial (or Gaussian) basis $\phi_j(x) = \exp\left(-\frac{(x-\mu_j)^2}{\ell^2}\right)$

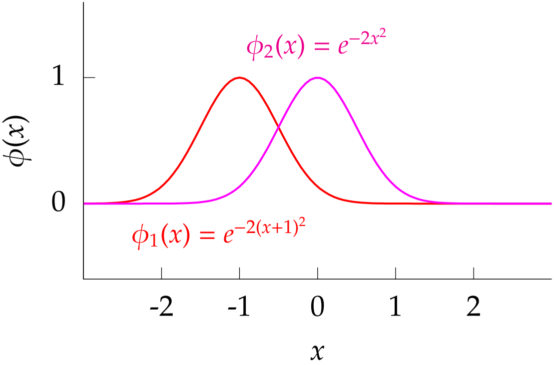


Figure: Radial basis functions.

Radial Basis Functions

- Or they can be local. E.g. radial (or Gaussian) basis $\phi_j(x) = \exp\left(-\frac{(x-\mu_j)^2}{\ell^2}\right)$

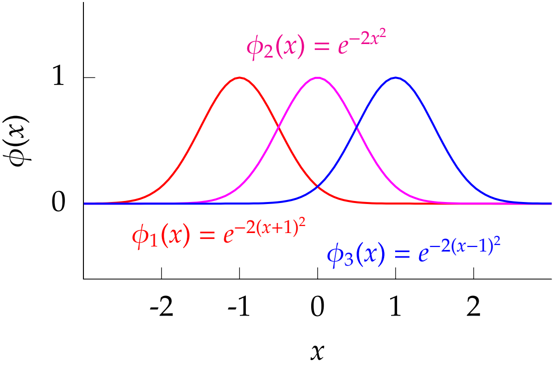


Figure: Radial basis functions.

Functions Derived from Radial Basis

$$f(x) = w_1 e^{-2(x+1)^2} + w_2 e^{-2x^2} + w_3 e^{-2(x-1)^2}$$

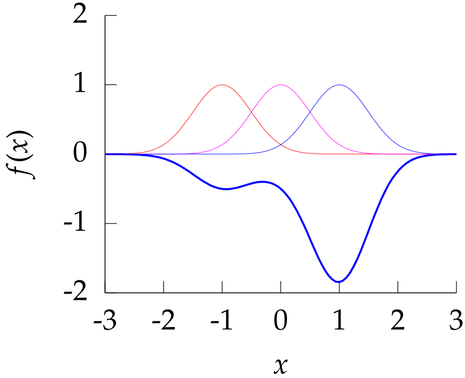


Figure: Function from radial basis with weights $w_1 = -0.47518$, $w_2 = -0.18924$, $w_3 = -1.8183$.

Functions Derived from Radial Basis

$$f(x) = w_1 e^{-2(x+1)^2} + w_2 e^{-2x^2} + w_3 e^{-2(x-1)^2}$$

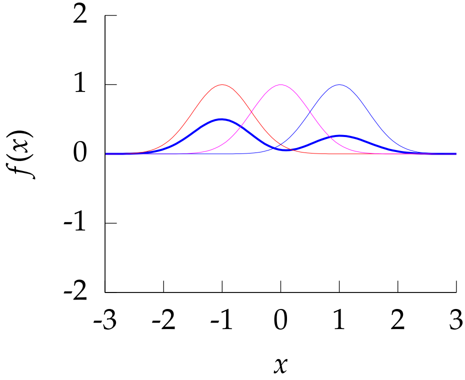


Figure: Function from radial basis with weights $w_1 = 0.50596$, $w_2 = -0.046315$, $w_3 = 0.26813$.

Functions Derived from Radial Basis

$$f(x) = w_1 e^{-2(x+1)^2} + w_2 e^{-2x^2} + w_3 e^{-2(x-1)^2}$$

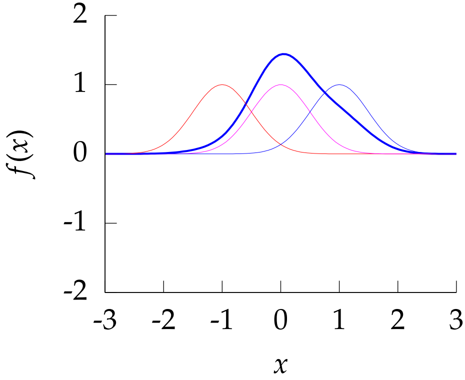


Figure: Function from radial basis with weights $w_1 = 0.07179$, $w_2 = 1.3591$, $w_3 = 0.50604$.

Probabilistic Model with Basis Functions

- Define a general function:

$$f(\mathbf{x}_i) = \mathbf{w}^\top \boldsymbol{\phi}(\mathbf{x}_i)$$

- Corrupt with independent noise:

$$y(\mathbf{x}_i) = f(\mathbf{x}_i) + \epsilon_i$$

$$\boldsymbol{\epsilon} \sim \prod_{i=1}^{n} \mathcal{N}\left(0, \sigma^2\right)$$

- Implies the following likelihood:

$$p(\mathbf{y}|\mathbf{w}, \sigma^2) = \prod_{i=1}^{n} \mathcal{N}\left(y_i|\mathbf{w}^\top \boldsymbol{\phi}(\mathbf{x}_i), \sigma^2\right)$$

Multivariate Regression Likelihood

- Noise corrupted data point

$$y_i = \mathbf{w}^\top \mathbf{x}_{i,:} + \epsilon_i$$

- Multivariate regression likelihood:

$$p(\mathbf{y}|\mathbf{X}, \mathbf{w}) = \frac{1}{(2\pi\sigma^2)^{n/2}} \exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n}\left(y_i - \mathbf{w}^\top \mathbf{x}_{i,:}\right)^2\right)$$

- Now use a multivariate Gaussian prior:

$$p(\mathbf{w}) = \frac{1}{(2\pi\alpha)^{\frac{p}{2}}} \exp\left(-\frac{1}{2\alpha}\mathbf{w}^\top \mathbf{w}\right)$$

Multivariate Regression Likelihood

- Noise corrupted data point

$$y_i = \mathbf{w}^\top \mathbf{x}_{i,:} + \epsilon_i$$

- Multivariate regression likelihood:

$$p(\mathbf{y}|\mathbf{X}, \mathbf{w}) = \frac{1}{(2\pi\sigma^2)^{n/2}} \exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n}\left(y_i - \mathbf{w}^\top \mathbf{x}_{i,:}\right)^2\right)$$

- Now use a multivariate Gaussian prior:

$$p(\mathbf{w}) = \frac{1}{(2\pi\alpha)^{\frac{p}{2}}} \exp\left(-\frac{1}{2\alpha}\mathbf{w}^\top \mathbf{w}\right)$$

Multivariate Regression Likelihood

- Noise corrupted data point

$$y_i = \mathbf{w}^\top \mathbf{x}_{i,:} + \epsilon_i$$

- Multivariate regression likelihood:

$$p(\mathbf{y}|\mathbf{X}, \mathbf{w}) = \frac{1}{(2\pi\sigma^2)^{n/2}} \exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n}\left(y_i - \mathbf{w}^\top \mathbf{x}_{i,:}\right)^2\right)$$

- Now use a multivariate Gaussian prior:

$$p(\mathbf{w}) = \frac{1}{(2\pi\alpha)^{\frac{p}{2}}} \exp\left(-\frac{1}{2\alpha}\mathbf{w}^\top \mathbf{w}\right)$$

Posterior Density

- Once again we want to know the posterior:

$$p(\mathbf{w}|\mathbf{y}, \mathbf{X}) \propto p(\mathbf{y}|\mathbf{X}, \mathbf{w})p(\mathbf{w})$$

- And we can compute by completing the square.

$$\begin{aligned}\log p(\mathbf{w}|\mathbf{y}, \mathbf{X}) = &-\frac{1}{2\sigma^2}\sum_{i=1}^{n} y_i^2 + \frac{1}{\sigma^2}\sum_{i=1}^{n} y_i \mathbf{x}_{i,:}^\top \mathbf{w} \\ &- \frac{1}{2\sigma^2}\sum_{i=1}^{n} \mathbf{w}^\top \mathbf{x}_{i,:}\mathbf{x}_{i,:}^\top \mathbf{w} - \frac{1}{2\alpha}\mathbf{w}^\top \mathbf{w} + \text{const.}\end{aligned}$$

Posterior Density

- Once again we want to know the posterior:

$$p(\mathbf{w}|\mathbf{y}, \mathbf{X}) \propto p(\mathbf{y}|\mathbf{X}, \mathbf{w})p(\mathbf{w})$$

- And we can compute by completing the square.

$$\begin{aligned}\log p(\mathbf{w}|\mathbf{y}, \mathbf{X}) = &- \frac{1}{2\sigma^2}\sum_{i=1}^{n} y_i^2 + \frac{1}{\sigma^2}\sum_{i=1}^{n} y_i \mathbf{x}_{i,:}^\top \mathbf{w} \\ &- \frac{1}{2\sigma^2}\sum_{i=1}^{n} \mathbf{w}^\top \mathbf{x}_{i,:}\mathbf{x}_{i,:}^\top \mathbf{w} - \frac{1}{2\alpha}\mathbf{w}^\top \mathbf{w} + \text{const.}\end{aligned}$$

Computing the Posterior

- By inspection we extract the inverse covariance

$$\log p(\mathbf{w}|\mathbf{y}, \mathbf{X}) = -\frac{1}{2\sigma^2}\sum_{i=1}^{n} y_i^2 + \frac{1}{\sigma^2}\sum_{i=1}^{n} y_i \mathbf{x}_{i,:}^\top \mathbf{w}$$
$$-\frac{1}{2\sigma^2}\sum_{i=1}^{n} \mathbf{w}^\top \mathbf{x}_{i,:}\mathbf{x}_{i,:}^\top \mathbf{w} - \frac{1}{2\alpha}\mathbf{w}^\top \mathbf{w} + \text{const.}$$

- Completing the square allows us to compute the mean.

Computing the Posterior

- By inspection we extract the inverse covariance

$$\begin{aligned}\log p(\mathbf{w}|\mathbf{y}, \mathbf{X}) = &-\frac{1}{2\sigma^2}\mathbf{y}^\top\mathbf{y} + \frac{1}{\sigma^2}\mathbf{y}^\top\mathbf{X}\mathbf{w} \\ &-\frac{1}{2\sigma^2}\mathbf{w}^\top\mathbf{X}^\top\mathbf{X}\mathbf{w} - \frac{1}{2\alpha}\mathbf{w}^\top\mathbf{w} + \text{const.}\end{aligned}$$

- Completing the square allows us to compute the mean.

Computing the Posterior

- By inspection we extract the inverse covariance

$$
\begin{aligned}
\log p(\mathbf{w}|\mathbf{y}, \mathbf{X}) = &- \frac{1}{2\sigma^2}\mathbf{y}^\top\mathbf{y} + \frac{1}{\sigma^2}\mathbf{y}^\top\mathbf{X}\mathbf{w} \\
&- \frac{1}{2}\mathbf{w}^\top\left[\sigma^{-1}\mathbf{X}^\top\mathbf{X} + \alpha^{-1}\mathbf{I}\right]\mathbf{w} + \text{const.}
\end{aligned}
$$

- Completing the square allows us to compute the mean.

Making Predictions

- Giving a Gaussian density

$$p(\mathbf{w}|\mathbf{y},\mathbf{X}) = \mathcal{N}\left(\mathbf{w}|\boldsymbol{\mu}_w, \mathbf{C}_w\right)$$

$$\mathbf{C}_w = \left[\sigma^{-2}\mathbf{X}^\top\mathbf{X} + \alpha^{-1}\mathbf{I}\right]^{-1} \qquad \boldsymbol{\mu}_w = \mathbf{C}_w\sigma^{-2}\mathbf{X}^\top\mathbf{y}$$

- Posterior is combined with 'test data' likelihood to make future predictions:

$$p(y_*|\mathbf{x}_*,\mathbf{X},\mathbf{y}) = \int p(y_*|\mathbf{x}_*,\mathbf{w})p(\mathbf{w}|\mathbf{X},\mathbf{y})\mathrm{d}\mathbf{w}$$

Bayesian vs Maximum Likelihood

- Note the similarity between posterior mean

$$\boldsymbol{\mu}_w = (\sigma^{-2}\mathbf{X}^\top\mathbf{X} + \alpha^{-1}\mathbf{I})^{-1}\sigma^{-2}\mathbf{X}^\top\mathbf{y}$$

- and Maximum likelihood solution

$$\hat{\mathbf{w}} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y}$$

Marginal Likelihood

- In some sense though the *real* model is now the marginal likelihood.
- Marginalization of $\mathbf{W}$ follows sum rule

  $$p(\mathbf{y}|\mathbf{X}) = \int p(\mathbf{y}|\mathbf{X}, \mathbf{w})p(\mathbf{w})\mathrm{d}\mathbf{w}$$

  giving

  $$p(\mathbf{y}|\mathbf{X}) = \mathcal{N}\left(\mathbf{y}|\mathbf{0}, \alpha\mathbf{X}\mathbf{X}^\top + \sigma^2\mathbf{I}\right)$$

- Often the integral is intractable.
- Leads to variational approximations, MCMC (Michael Betancourt, Mark Girolami), Laplace approximation (Harvard Rue).
- For the case of Gaussians it's trivial!!

Marginal Likelihood

- Can compute the marginal likelihood as:

$$p(\mathbf{y}|\mathbf{X}, \alpha, \sigma) = \mathcal{N}\left(\mathbf{y}|\mathbf{0}, \alpha\mathbf{X}\mathbf{X}^\top + \sigma^2\mathbf{I}\right)$$

- Or if we use a basis set we have

$$p(\mathbf{y}|\mathbf{X}, \alpha, \sigma) = \mathcal{N}\left(\mathbf{y}|\mathbf{0}, \alpha\boldsymbol{\Phi}\boldsymbol{\Phi}^\top + \sigma^2\mathbf{I}\right)$$

- This Gaussian is no longer i.i.d. across data and *this* is where things get interesting.

Marginal Likelihood

- The marginal likelihood can also be computed, it has the form:

$$p(\mathbf{y}|\mathbf{X}, \sigma^2, \alpha) = \frac{1}{(2\pi)^{\frac{n}{2}} |\mathbf{K}|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}\mathbf{y}^\top \mathbf{K}^{-1}\mathbf{y}\right)$$

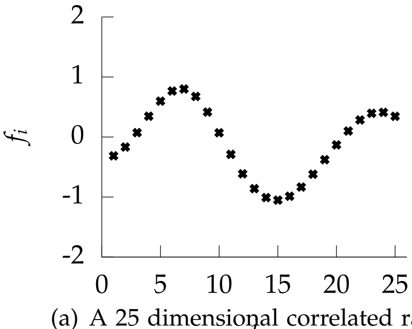
  where $\mathbf{K} = \alpha\mathbf{\Phi}\mathbf{\Phi}^\top + \sigma^2\mathbf{I}$.
- So it is a zero mean n-dimensional Gaussian with covariance matrix $\mathbf{K}$.

Sampling a Function

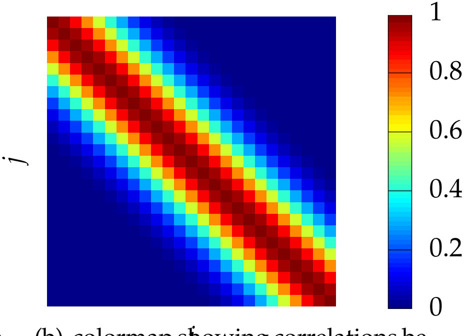
Multi-variate Gaussians

- We will consider a Gaussian with a particular structure of covariance matrix.
- Generate a single sample from this 25 dimensional Gaussian distribution, $\mathbf{f} = [f_1, f_2 \ldots f_{25}]$.
- We will plot these points against their index.

Gaussian Distribution Sample



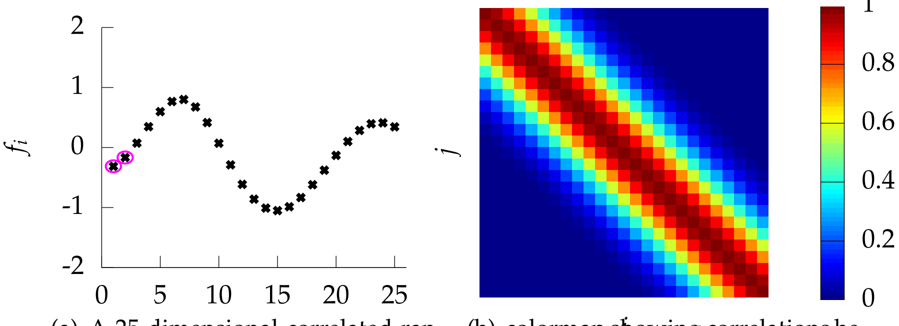
(a) A 25 dimensional correlated random variable (values ploted against index)



(b) colormap showing correlations between dimensions.

Figure: A sample from a 25 dimensional Gaussian distribution.

Gaussian Distribution Sample

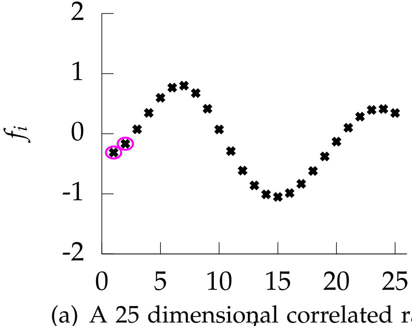


(a) A 25 dimensional correlated random variable (values ploted against index)

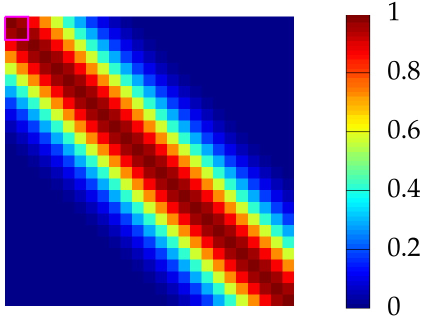
(b) colormap showing correlations between dimensions.

Figure: A sample from a 25 dimensional Gaussian distribution.

Gaussian Distribution Sample



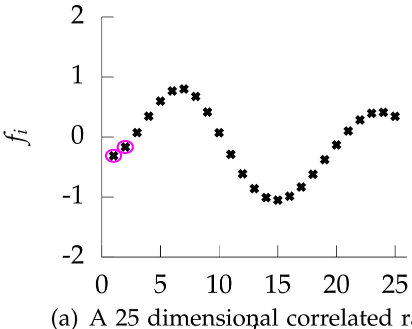
(a) A 25 dimensional correlated random variable (values ploted against index)



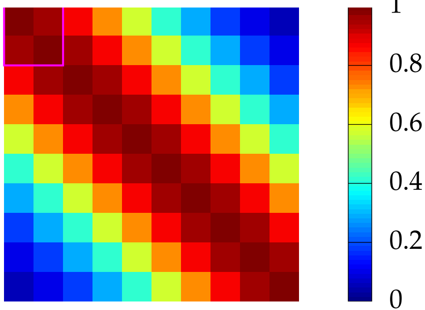
(b) colormap showing correlations between dimensions.

Figure: A sample from a 25 dimensional Gaussian distribution.

Gaussian Distribution Sample



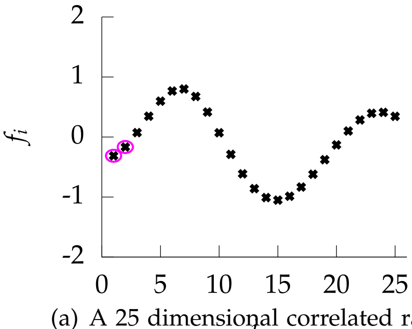
(a) A 25 dimensional correlated random variable (values ploted against index)



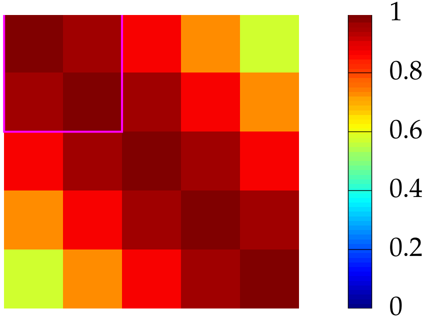
(b) colormap showing correlations between dimensions.

Figure: A sample from a 25 dimensional Gaussian distribution.

Gaussian Distribution Sample



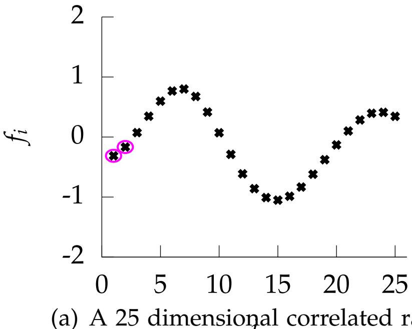
(a) A 25 dimensional correlated random variable (values ploted against index)



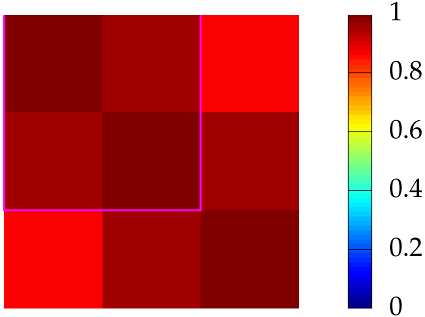
(b) colormap showing correlations between dimensions.

Figure: A sample from a 25 dimensional Gaussian distribution.

Gaussian Distribution Sample



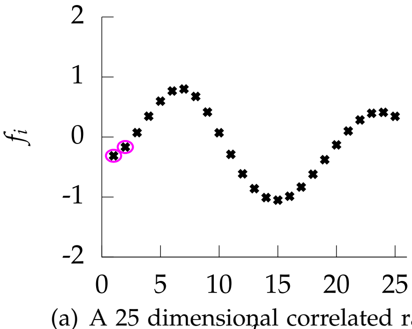
(a) A 25 dimensional correlated random variable (values ploted against index)



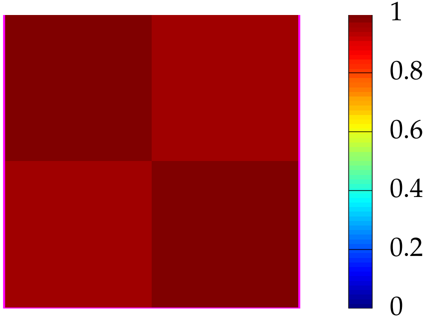
(b) colormap showing correlations between dimensions.

Figure: A sample from a 25 dimensional Gaussian distribution.

Gaussian Distribution Sample



(a) A 25 dimensional correlated random variable (values ploted against index)



(b) colormap showing correlations between dimensions.

Figure: A sample from a 25 dimensional Gaussian distribution.

Computing the Expected Output

- Given the posterior for the parameters, how can we compute the expected output at a given location?
- Output of model at location $\mathbf{x}_i$ is given by

$$f(\mathbf{x}_i; \mathbf{w}) = \boldsymbol{\phi}_i^\top \mathbf{w}$$

- We want the expected output under the posterior density, $p(\mathbf{w}|\mathbf{y}, \mathbf{X}, \sigma^2, \alpha)$.
- Mean of mapping function will be given by

$$\begin{aligned}\left\langle f(\mathbf{x}_i; \mathbf{w})\right\rangle_{p(\mathbf{w}|\mathbf{y},\mathbf{X},\sigma^2,\alpha)} &= \boldsymbol{\phi}_i^\top \left\langle \mathbf{w}\right\rangle_{p(\mathbf{w}|\mathbf{y},\mathbf{X},\sigma^2,\alpha)} \\ &= \boldsymbol{\phi}_i^\top \boldsymbol{\mu}_w\end{aligned}$$

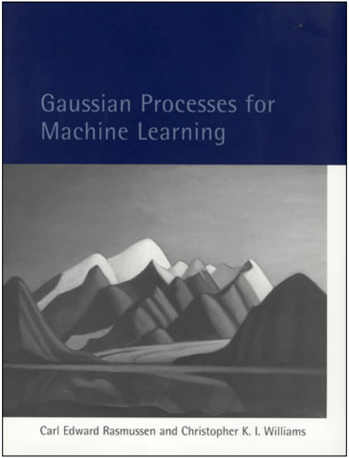
Variance of Expected Output

- Variance of model at location $\mathbf{x}_i$ is given by

$$\begin{aligned}\mathrm{var}(f(\mathbf{x}_i;\mathbf{w})) &= \left\langle (f(\mathbf{x}_i;\mathbf{w}))^2 \right\rangle - \left\langle f(\mathbf{x}_i;\mathbf{w}) \right\rangle^2 \\ &= \boldsymbol{\phi}_i^\top \left\langle \mathbf{w}\mathbf{w}^\top \right\rangle \boldsymbol{\phi}_i - \boldsymbol{\phi}_i^\top \left\langle \mathbf{w} \right\rangle \left\langle \mathbf{w} \right\rangle^\top \boldsymbol{\phi}_i \\ &= \boldsymbol{\phi}_i^\top \mathbf{C}_w \boldsymbol{\phi}_i\end{aligned}$$

  where all these expectations are taken under the posterior density, $p(\mathbf{w}|\mathbf{y}, \mathbf{X}, \sigma^2, \alpha)$.

Book

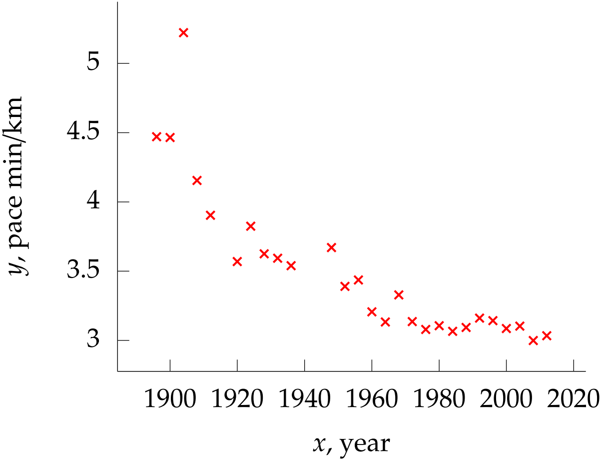

Gaussian Processes for
Machine Learning
Carl Edward Rasmussen and Christopher K. I. Williams

Olympic Marathon Data

- Gold medal times for Olympic Marathon since 1896.
- Marathons before 1924 didn't have a standardised distance.
- Present results using pace per km.
- In 1904 Marathon was badly organised leading to very slow times.

Image from Wikimedia Commons
`http://bit.ly/16kMKHQ`

Olympic Marathon Data



Olympic Marathon Data.

Olympics Data analysis

- Use Bayesian approach on olympics data with polynomials.
- Choose a prior $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \alpha\mathbf{I})$ with $\alpha = 1$.
- Choose noise variance $\sigma^2 = 0.01$

Sampling the Prior

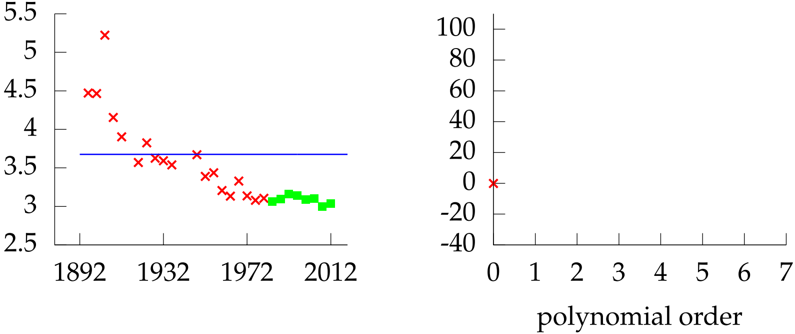
- Always useful to perform a 'sanity check' and sample from the prior before observing the data.
- Since $\mathbf{y} = \mathbf{\Phi}\mathbf{w} + \boldsymbol{\epsilon}$ just need to sample

$$w \sim \mathcal{N}(0, \alpha)$$

$$\boldsymbol{\epsilon} \sim \mathcal{N}\left(\mathbf{0}, \sigma^2\right)$$

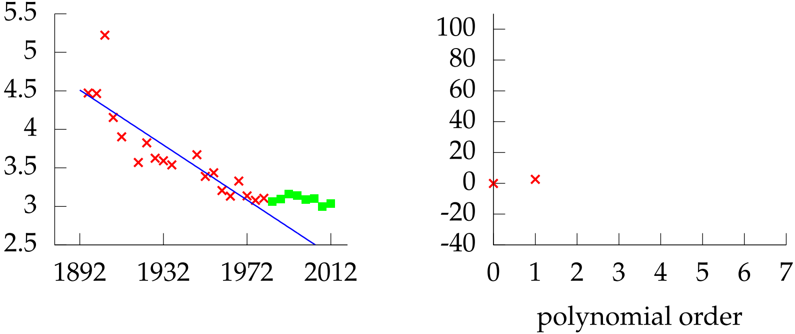
with $\alpha = 1$ and $\boldsymbol{\epsilon} = 0.01$.

Recall: Validation Set for Maximum Likelihood



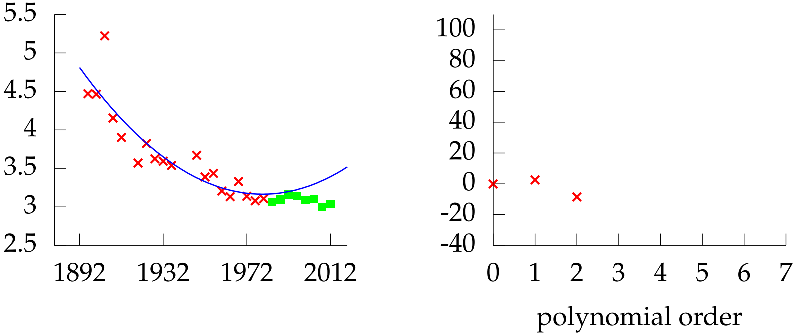
Left: fit to data, *Right*: model error. Polynomial order 0, training error -1.8774, validation error -0.13132, $\sigma^2 = 0.302$, $\sigma = 0.549$.

Recall: Validation Set for Maximum Likelihood



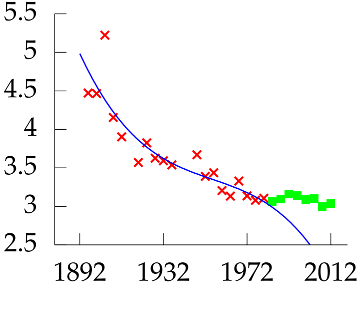
Left: fit to data, *Right*: model error. Polynomial order 1, training error -15.325, validation error 2.5863, $\sigma^2 = 0.0733$, $\sigma = 0.271$.

Recall: Validation Set for Maximum Likelihood



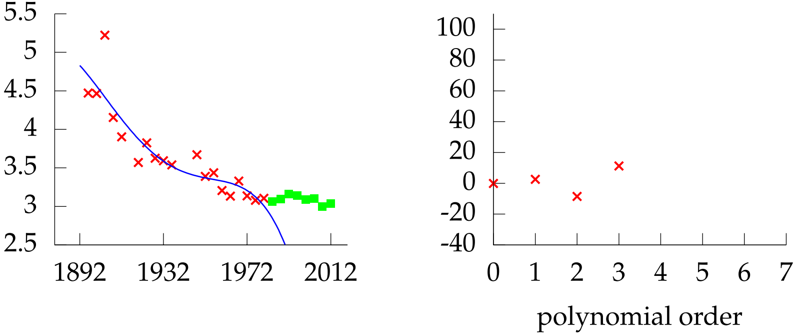
Left: fit to data, *Right*: model error. Polynomial order 2, training error -17.579, validation error -8.4831, $\sigma^2 = 0.0578$, $\sigma = 0.240$.

Recall: Validation Set for Maximum Likelihood



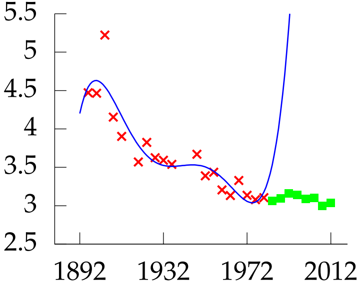
Left: fit to data, *Right*: model error. Polynomial order 3, training error -18.064, validation error 11.27, $\sigma^2 = 0.0549$, $\sigma = 0.234$.

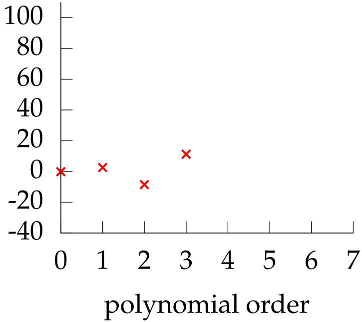
Recall: Validation Set for Maximum Likelihood



Left: fit to data, *Right*: model error. Polynomial order 4, training error -18.245, validation error 232.92, $\sigma^2 = 0.0539$, $\sigma = 0.232$.

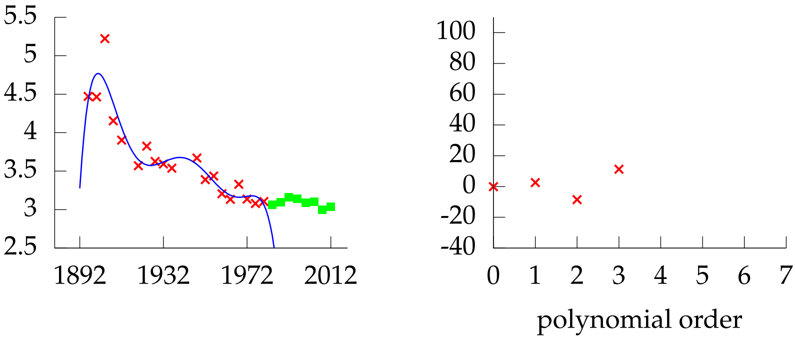
Recall: Validation Set for Maximum Likelihood





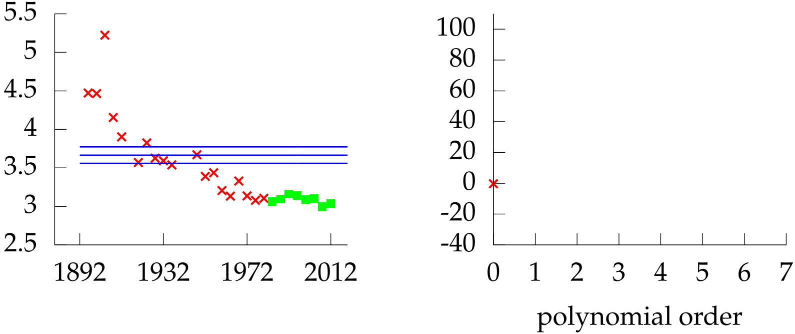
Left: fit to data, *Right*: model error. Polynomial order 5, training error -20.471, validation error 9898.1, $\sigma^2 = 0.0426$, $\sigma = 0.207$.

Recall: Validation Set for Maximum Likelihood



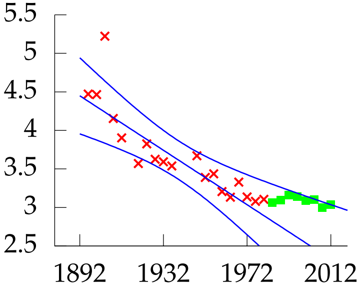
Left: fit to data, *Right*: model error. Polynomial order 6, training error -22.881, validation error 67775, $\sigma^2 = 0.0331$, $\sigma = 0.182$.

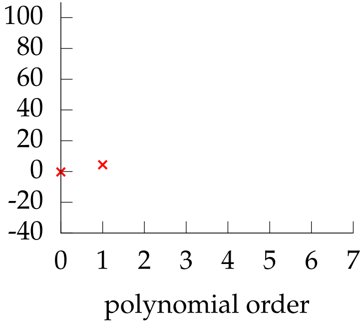
Validation Set



Left: fit to data, *Right*: model error. Polynomial order 0, training error 29.757, validation error -0.29243, $\sigma^2 = 0.302$, $\sigma = 0.550$.

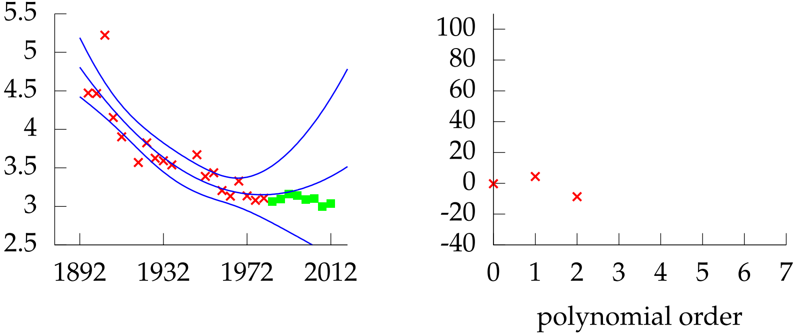
Validation Set





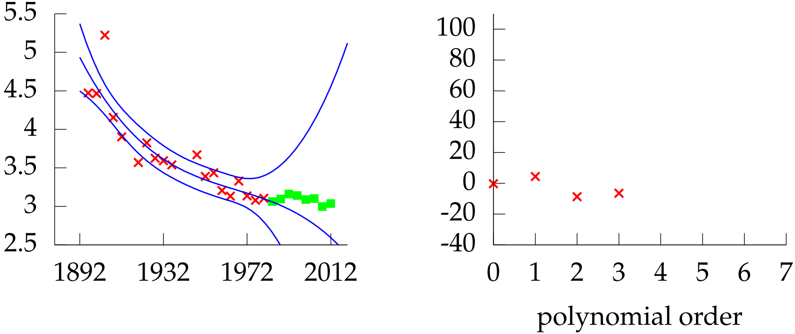
Left: fit to data, *Right*: model error. Polynomial order 1, training error 14.942, validation error 4.4027, $\sigma^2 = 0.0762$, $\sigma = 0.276$.

Validation Set



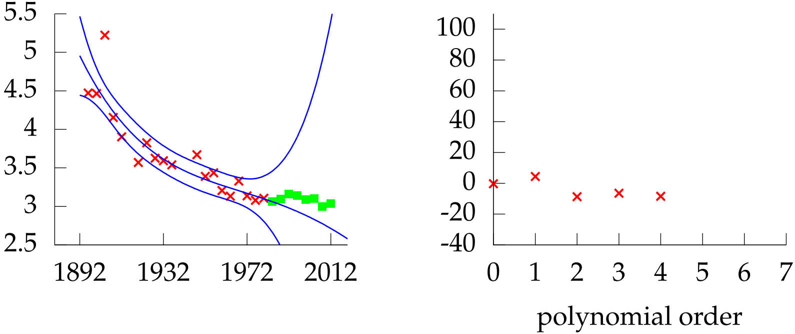
Left: fit to data, *Right*: model error. Polynomial order 2, training error 9.7206, validation error -8.6623, $\sigma^2 = 0.0580$, $\sigma = 0.241$.

Validation Set



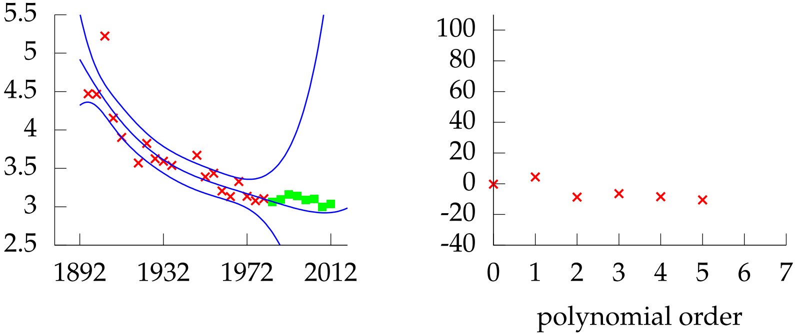
Left: fit to data, *Right*: model error. Polynomial order 3, training error 10.416, validation error -6.4726, $\sigma^2 = 0.0555$, $\sigma = 0.236$.

Validation Set



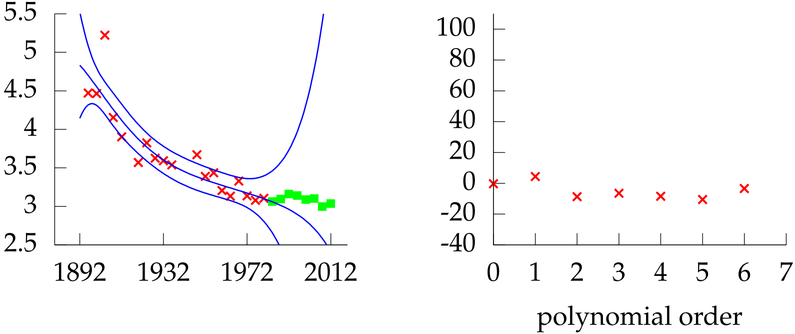
Left: fit to data, *Right*: model error. Polynomial order 4, training error 11.34, validation error -8.431, $\sigma^2 = 0.0555$, $\sigma = 0.236$.

Validation Set



Left: fit to data, *Right*: model error. Polynomial order 5, training error 11.986, validation error -10.483, $\sigma^2 = 0.0551$, $\sigma = 0.235$.

Validation Set



Left: fit to data, *Right*: model error. Polynomial order 6, training error 12.369, validation error -3.3823, $\sigma^2 = 0.0537$, $\sigma = 0.232$.

Regularized Mean

- Validation fit here based on mean solution for $\mathbf{w}$ only.
- For Bayesian solution

$$\boldsymbol{\mu}_w = \left[\sigma^{-2}\boldsymbol{\Phi}^\top\boldsymbol{\Phi} + \alpha^{-1}\mathbf{I}\right]^{-1}\sigma^{-2}\boldsymbol{\Phi}^\top\mathbf{y}$$

  instead of

$$\mathbf{w}^* = \left[\boldsymbol{\Phi}^\top\boldsymbol{\Phi}\right]^{-1}\boldsymbol{\Phi}^\top\mathbf{y}$$

- Two are equivalent when $\alpha \to \infty$.
- Equivalent to a prior for $\mathbf{w}$ with infinite variance.
- In other cases $\alpha\mathbf{I}$ *regularizes* the system (keeps parameters smaller).

Sampling the Posterior

- Now check samples by extracting $\mathbf{w}$ from the *posterior*.
- Now for $\mathbf{y} = \mathbf{\Phi}\mathbf{w} + \boldsymbol{\epsilon}$ need

$$w \sim \mathcal{N}\left(\boldsymbol{\mu}_w, \mathbf{C}_w\right)$$

with $\mathbf{C}_w = \left[\sigma^{-2}\mathbf{\Phi}^\top\mathbf{\Phi} + \alpha^{-1}\mathbf{I}\right]^{-1}$ and $\boldsymbol{\mu}_w = \mathbf{C}_w\sigma^{-2}\mathbf{\Phi}^\top\mathbf{y}$

$$\boldsymbol{\epsilon} \sim \mathcal{N}\left(\mathbf{0}, \sigma^2\right)$$

with $\alpha = 1$ and $\epsilon = 0.01$.

Direct Construction of Covariance Matrix

Use matrix notation to write function,

$$f(\mathbf{x}_i; \mathbf{w}) = \sum_{k=1}^{m} w_k \phi_k(\mathbf{x}_i)$$

Direct Construction of Covariance Matrix

Use matrix notation to write function,

$$f(\mathbf{x}_i; \mathbf{w}) = \sum_{k=1}^{m} w_k \phi_k(\mathbf{x}_i)$$

computed at training data gives a vector

$$\mathbf{f} = \mathbf{\Phi} \mathbf{w}.$$

Direct Construction of Covariance Matrix

Use matrix notation to write function,

$$f(\mathbf{x}_i; \mathbf{w}) = \sum_{k=1}^{m} w_k \phi_k(\mathbf{x}_i)$$

computed at training data gives a vector

$$\mathbf{f} = \boldsymbol{\Phi}\mathbf{w}.$$

$$\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \alpha\mathbf{I})$$

Direct Construction of Covariance Matrix

Use matrix notation to write function,

$$f(\mathbf{x}_i; \mathbf{w}) = \sum_{k=1}^{m} w_k \phi_k(\mathbf{x}_i)$$

computed at training data gives a vector

$$\mathbf{f} = \boldsymbol{\Phi}\mathbf{w}.$$

$$\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \alpha\mathbf{I})$$

$\mathbf{w}$ and $\mathbf{f}$ are only related by an *inner product*.

Direct Construction of Covariance Matrix

Use matrix notation to write function,

$$f(\mathbf{x}_i; \mathbf{w}) = \sum_{k=1}^{m} w_k \phi_k(\mathbf{x}_i)$$

computed at training data gives a vector

$$\mathbf{f} = \mathbf{\Phi}\mathbf{w}.$$

$$\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \alpha \mathbf{I})$$

$\mathbf{w}$ and $\mathbf{f}$ are only related by an *inner product*.

$\mathbf{\Phi} \in \Re^{n \times p}$ is a *design matrix*

Direct Construction of Covariance Matrix

Use matrix notation to write function,

$$f(\mathbf{x}_i; \mathbf{w}) = \sum_{k=1}^{m} w_k \phi_k(\mathbf{x}_i)$$

computed at training data gives a vector

$$\mathbf{f} = \mathbf{\Phi}\mathbf{w}.$$

$$\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \alpha\mathbf{I})$$

$\mathbf{w}$ and $\mathbf{f}$ are only related by an *inner product*.

$\mathbf{\Phi} \in \Re^{n \times p}$ is a *design matrix*

$\mathbf{\Phi}$ is fixed and non-stochastic for a given training set.

Direct Construction of Covariance Matrix

Use matrix notation to write function,

$$f(\mathbf{x}_i; \mathbf{w}) = \sum_{k=1}^{m} w_k \phi_k(\mathbf{x}_i)$$

computed at training data gives a vector

$$\mathbf{f} = \mathbf{\Phi}\mathbf{w}.$$

$$\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \alpha\mathbf{I})$$

$\mathbf{w}$ and $\mathbf{f}$ are only related by an *inner product*.

$\mathbf{\Phi} \in \Re^{n\times p}$ is a *design matrix*

$\mathbf{\Phi}$ is fixed and non-stochastic for a given training set.

$\mathbf{f}$ is Gaussian distributed.

Expectations

- We have
$$\langle \mathbf{f} \rangle = \mathbf{\Phi} \langle \mathbf{w} \rangle .$$
- Prior mean of $\mathbf{w}$ was zero giving
$$\langle \mathbf{f} \rangle = \mathbf{0}.$$
- Prior covariance of $\mathbf{f}$ is
$$\mathbf{K} = \left\langle \mathbf{f}\mathbf{f}^\top \right\rangle - \langle \mathbf{f} \rangle \langle \mathbf{f} \rangle^\top$$

We use $\langle\cdot\rangle$ to denote expectations under prior distributions.

Expectations

- We have

$$\langle \mathbf{f} \rangle = \mathbf{\Phi} \langle \mathbf{w} \rangle .$$

- Prior mean of $\mathbf{w}$ was zero giving

$$\langle \mathbf{f} \rangle = \mathbf{0}.$$

- Prior covariance of $\mathbf{f}$ is

$$\mathbf{K} = \left\langle \mathbf{f}\mathbf{f}^\top \right\rangle - \langle \mathbf{f} \rangle \langle \mathbf{f} \rangle^\top$$

We use $\langle \cdot \rangle$ to denote expectations under prior distributions.

Expectations

- We have
$$\langle \mathbf{f} \rangle = \mathbf{\Phi} \langle \mathbf{w} \rangle .$$
- Prior mean of $\mathbf{w}$ was zero giving
$$\langle \mathbf{f} \rangle = \mathbf{0}.$$
- Prior covariance of $\mathbf{f}$ is
$$\mathbf{K} = \left\langle \mathbf{f}\mathbf{f}^\top \right\rangle - \langle \mathbf{f} \rangle \langle \mathbf{f} \rangle^\top$$

We use $\langle \cdot \rangle$ to denote expectations under prior distributions.

Expectations

- We have
$$\langle \mathbf{f} \rangle = \boldsymbol{\Phi} \langle \mathbf{w} \rangle .$$
- Prior mean of $\mathbf{w}$ was zero giving
$$\langle \mathbf{f} \rangle = \mathbf{0}.$$
- Prior covariance of $\mathbf{f}$ is
$$\mathbf{K} = \left\langle \mathbf{f}\mathbf{f}^\top \right\rangle - \langle \mathbf{f} \rangle \langle \mathbf{f} \rangle^\top$$
$$\left\langle \mathbf{f}\mathbf{f}^\top \right\rangle = \boldsymbol{\Phi} \left\langle \mathbf{w}\mathbf{w}^\top \right\rangle \boldsymbol{\Phi}^\top,$$
giving
$$\mathbf{K} = \alpha \boldsymbol{\Phi}\boldsymbol{\Phi}^\top.$$

We use $\langle \cdot \rangle$ to denote expectations under prior distributions.

Covariance between Two Points

- The prior covariance between two points $\mathbf{x}_i$ and $\mathbf{x}_j$ is

$$k\left(\mathbf{x}_i, \mathbf{x}_j\right) = \alpha \phi_{:}\left(\mathbf{x}_i\right)^\top \phi_{:}\left(\mathbf{x}_j\right),$$

  or in sum notation

$$k\left(\mathbf{x}_i, \mathbf{x}_j\right) = \alpha \sum_{k=1}^{m} \phi_k\left(\mathbf{x}_i\right) \phi_k\left(\mathbf{x}_j\right)$$

- For the radial basis used this gives

$$k\left(\mathbf{x}_i, \mathbf{x}_j\right) = \alpha \sum_{k=1}^{m} \exp\left(-\frac{\left|\mathbf{x}_i - \boldsymbol{\mu}_k\right|^2 + \left|\mathbf{x}_j - \boldsymbol{\mu}_k\right|^2}{2\ell^2}\right).$$

Covariance between Two Points

- The prior covariance between two points $\mathbf{x}_i$ and $\mathbf{x}_j$ is

$$k\left(\mathbf{x}_i, \mathbf{x}_j\right) = \alpha \phi_{:}\left(\mathbf{x}_i\right)^{\top} \phi_{:}\left(\mathbf{x}_j\right),$$

  or in sum notation

$$k\left(\mathbf{x}_i, \mathbf{x}_j\right) = \alpha \sum_{k=1}^{m} \phi_k\left(\mathbf{x}_i\right) \phi_k\left(\mathbf{x}_j\right)$$

- For the radial basis used this gives

$$k\left(\mathbf{x}_i, \mathbf{x}_j\right) = \alpha \sum_{k=1}^{m} \exp\left(-\frac{\left|\mathbf{x}_i - \boldsymbol{\mu}_k\right|^2 + \left|\mathbf{x}_j - \boldsymbol{\mu}_k\right|^2}{2\ell^2}\right).$$

Covariance between Two Points

- The prior covariance between two points $\mathbf{x}_i$ and $\mathbf{x}_j$ is

$$k\left(\mathbf{x}_i, \mathbf{x}_j\right) = \alpha \phi_{:}\left(\mathbf{x}_i\right)^\top \phi_{:}\left(\mathbf{x}_j\right),$$

  or in sum notation

$$k\left(\mathbf{x}_i, \mathbf{x}_j\right) = \alpha \sum_{k=1}^{m} \phi_k\left(\mathbf{x}_i\right) \phi_k\left(\mathbf{x}_j\right)$$

- For the radial basis used this gives

$$k\left(\mathbf{x}_i, \mathbf{x}_j\right) = \alpha \sum_{k=1}^{m} \exp\left(-\frac{\left|\mathbf{x}_i - \boldsymbol{\mu}_k\right|^2 + \left|\mathbf{x}_j - \boldsymbol{\mu}_k\right|^2}{2\ell^2}\right).$$

Covariance between Two Points

- The prior covariance between two points $\mathbf{x}_i$ and $\mathbf{x}_j$ is

$$k\left(\mathbf{x}_i, \mathbf{x}_j\right) = \alpha \phi_{:}\left(\mathbf{x}_i\right)^\top \phi_{:}\left(\mathbf{x}_j\right),$$

  or in sum notation

$$k\left(\mathbf{x}_i, \mathbf{x}_j\right) = \alpha \sum_{k=1}^{m} \phi_k\left(\mathbf{x}_i\right) \phi_k\left(\mathbf{x}_j\right)$$

- For the radial basis used this gives

$$k\left(\mathbf{x}_i, \mathbf{x}_j\right) = \alpha \sum_{k=1}^{m} \exp\left(-\frac{\left|\mathbf{x}_i - \boldsymbol{\mu}_k\right|^2 + \left|\mathbf{x}_j - \boldsymbol{\mu}_k\right|^2}{2\ell^2}\right).$$

Covariance Functions and Mercer Kernels

- Mercer Kernels and Covariance Functions are similar.
- the kernel perspective does not make a probabilistic interpretation of the covariance function.
- Algorithms can be simpler, but probabilistic interpretation is crucial for kernel parameter optimization.

Covariance Functions and Mercer Kernels

- Mercer Kernels and Covariance Functions are similar.
- the kernel perspective does not make a probabilistic interpretation of the covariance function.
- Algorithms can be simpler, but probabilistic interpretation is crucial for kernel parameter optimization.

Covariance Functions and Mercer Kernels

- Mercer Kernels and Covariance Functions are similar.
- the kernel perspective does not make a probabilistic interpretation of the covariance function.
- Algorithms can be simpler, but probabilistic interpretation is crucial for kernel parameter optimization.

More on Mercer Kernels

Let X be a metric space and $K : X \times X \to \Re$ a continuous and symmetric function. If we assume that K is positive definite, that is, for any set $\mathbf{x} = \{\mathbf{x}_1, \cdots, \mathbf{x}_n\} \subset X$ the $n \times n$ matrix $\mathbf{K}$ with components

$$\mathbf{K}_{ij} = K(\mathbf{x}_i, \mathbf{x}_j),$$

is positive semi-definite, then $\mathbf{K}$ is a Mercer kernel.

More on Mercer Kernels

Mercer's Theorem (1909): Let $K : X \times X \longrightarrow \Re$ a Mercer's kernel. Let λ_j the j-th eigenvalue of L_K and $\{\phi_j\}_{j \geq 1}$ the corresponding eigenvector. Then, for all $\mathbf{x}, \mathbf{x}' \in X$

$$K(\mathbf{x}, \mathbf{y}) = \sum_{j=1}^{\infty} \lambda_j \phi_j(\mathbf{x}) \phi_j(\mathbf{y})$$

where the convergence is absolute (for each $(\mathbf{x}, \mathbf{x}') \in X \times X$) and uniform (on $(\mathbf{x}, \mathbf{x}') \in X \times X$).

By using directly a kernel we are using a basis function implicitly (possibly with infinity elements: Bayesian non-parametrics).

Prediction with Correlated Gaussians

- Prediction of $\mathbf{f}_*$ from $\mathbf{f}$ requires multivariate *conditional density*.
- Multivariate conditional density is *also* Gaussian.

$$p(\mathbf{f}_*|\mathbf{f}) = \mathcal{N}\left(\mathbf{f}_*|\mathbf{K}_{*,\mathbf{f}}\mathbf{K}_{\mathbf{f},\mathbf{f}}^{-1}\mathbf{f}, \mathbf{K}_{*,*} - \mathbf{K}_{*,\mathbf{f}}\mathbf{K}_{\mathbf{f},\mathbf{f}}^{-1}\mathbf{K}_{\mathbf{f},*}\right)$$

- Here covariance of joint density is given by

$$\mathbf{K} = \begin{bmatrix}\mathbf{K}_{\mathbf{f},\mathbf{f}} & \mathbf{K}_{*,\mathbf{f}}\\ \mathbf{K}_{\mathbf{f},*} & \mathbf{K}_{*,*}\end{bmatrix}$$

Prediction with Correlated Gaussians

- Prediction of $\mathbf{f}_*$ from $\mathbf{f}$ requires multivariate *conditional density*.
- Multivariate conditional density is *also* Gaussian.

$$p(\mathbf{f}_*|\mathbf{f}) = \mathcal{N}\left(\mathbf{f}_*|\boldsymbol{\mu}, \boldsymbol{\Sigma}\right)$$

$$\boldsymbol{\mu} = \mathbf{K}_{*,\mathbf{f}}\mathbf{K}_{\mathbf{f},\mathbf{f}}^{-1}\mathbf{f}$$

$$\boldsymbol{\Sigma} = \mathbf{K}_{*,*} - \mathbf{K}_{*,\mathbf{f}}\mathbf{K}_{\mathbf{f},\mathbf{f}}^{-1}\mathbf{K}_{\mathbf{f},*}$$

- Here covariance of joint density is given by

$$\mathbf{K} = \begin{bmatrix} \mathbf{K}_{\mathbf{f},\mathbf{f}} & \mathbf{K}_{*,\mathbf{f}} \\ \mathbf{K}_{\mathbf{f},*} & \mathbf{K}_{*,*} \end{bmatrix}$$

Constructing Covariance Functions

- Sum of two covariances is also a covariance function.

$$k(\mathbf{x}, \mathbf{x}') = k_1(\mathbf{x}, \mathbf{x}') + k_2(\mathbf{x}, \mathbf{x}')$$

Constructing Covariance Functions

- Product of two covariances is also a covariance function.

$$k(\mathbf{x}, \mathbf{x}') = k_1(\mathbf{x}, \mathbf{x}')k_2(\mathbf{x}, \mathbf{x}')$$

Multiply by Deterministic Function

- If $f(\mathbf{x})$ is a Gaussian process.
- $g(\mathbf{x})$ is a deterministic function.
- $h(\mathbf{x}) = f(\mathbf{x})g(\mathbf{x})$
- Then
$$k_h(\mathbf{x}, \mathbf{x}') = g(\mathbf{x})k_f(\mathbf{x}, \mathbf{x}')g(\mathbf{x}')$$
where k_h is covariance for $h(\cdot)$ and k_f is covariance for $f(\cdot)$.

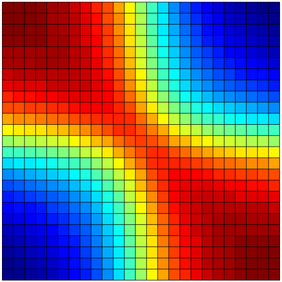
MLP Covariance Function

$$k(\mathbf{x}, \mathbf{x}') = \alpha \mathrm{asin}\left(\frac{w\mathbf{x}^\top \mathbf{x}' + b}{\sqrt{w\mathbf{x}^\top \mathbf{x} + b + 1}\sqrt{w\mathbf{x}'^\top \mathbf{x}' + b + 1}}\right)$$

- Based on infinite neural network model.

$$w = 40$$

$$b = 4$$



Covariance Functions

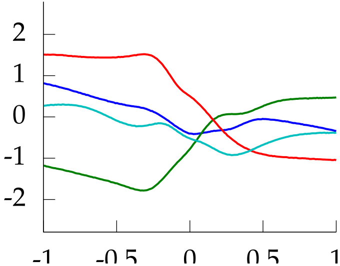
MLP Covariance Function

$$k(\mathbf{x}, \mathbf{x}') = \alpha \mathrm{asin}\left(\frac{w\mathbf{x}^\top \mathbf{x}' + b}{\sqrt{w\mathbf{x}^\top \mathbf{x} + b + 1}\sqrt{w\mathbf{x}'^\top \mathbf{x}' + b + 1}}\right)$$

- Based on infinite neural network model.

$$w = 40$$

$$b = 4$$



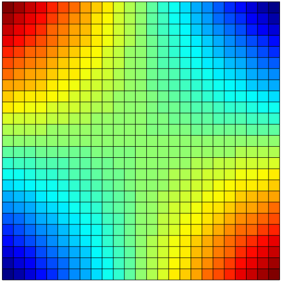
Covariance Functions

Linear Covariance Function

$$k(\mathbf{x}, \mathbf{x}') = \alpha \mathbf{x}^\top \mathbf{x}'$$

- Bayesian linear regression.

$$\alpha = 1$$



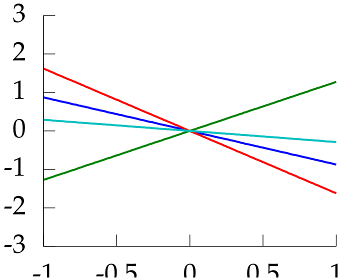
Covariance Functions

Linear Covariance Function

$$k(\mathbf{x}, \mathbf{x}') = \alpha \mathbf{x}^\top \mathbf{x}'$$

- Bayesian linear regression.

$$\alpha = 1$$



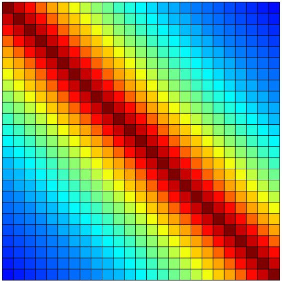
Covariance Functions

Where did this covariance matrix come from?

Ornstein-Uhlenbeck (stationary Gauss-Markov) covariance function

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

- In one dimension arises from a stochastic differential equation. Brownian motion in a parabolic tube.
- In higher dimension a Fourier filter of the form $\frac{1}{\pi(1+x^2)}$.



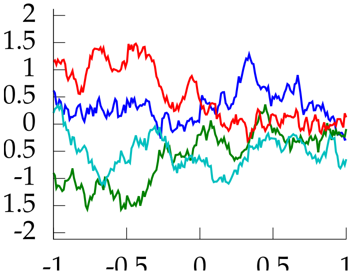
Covariance Functions

Where did this covariance matrix come from?

Ornstein-Uhlenbeck (stationary Gauss-Markov) covariance function

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

- In one dimension arises from a stochastic differential equation. Brownian motion in a parabolic tube.
- In higher dimension a Fourier filter of the form $\frac{1}{\pi(1+x^2)}$.



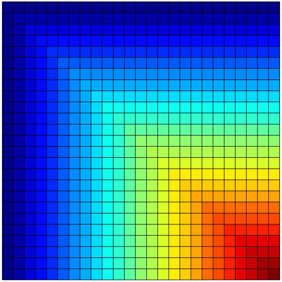
Covariance Functions

Where did this covariance matrix come from?

Markov Process

$$k(t, t') = \alpha \min(t, t')$$

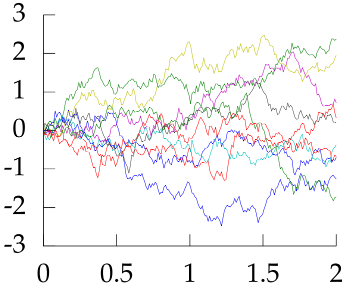
- Covariance matrix is built using the *inputs* to the function t.



Markov Process

$$k(t, t') = \alpha \min(t, t')$$

- Covariance matrix is built using the *inputs* to the function t.



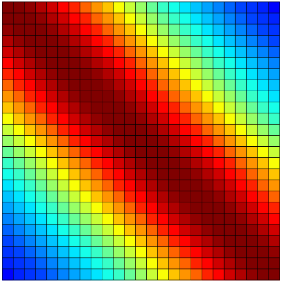
Covariance Functions

Where did this covariance matrix come from?

Matern 5/2 Covariance Function

$$k(\mathbf{x}, \mathbf{x}') = \alpha \left(1 + \sqrt{5} r + \frac{5}{3} r^2\right) \exp\left(-\sqrt{5} r\right) \quad \text{where} \quad r = \frac{\|\mathbf{x} - \mathbf{x}'\|_2}{\ell}$$

- Matern 5/2 is a twice differentiable covariance.
- Matern family constructed with Student-t filters in Fourier space.



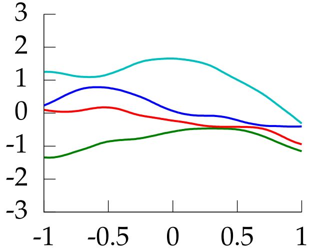
Covariance Functions

Where did this covariance matrix come from?

Matern 5/2 Covariance Function

$$k(\mathbf{x}, \mathbf{x}') = \alpha \left(1 + \sqrt{5}r + \frac{5}{3}r^2\right) \exp\left(-\sqrt{5}r\right) \quad \text{where} \quad r = \frac{\|\mathbf{x} - \mathbf{x}'\|_2}{\ell}$$

- Matern 5/2 is a twice differentiable covariance.
- Matern family constructed with Student-*t* filters in Fourier space.



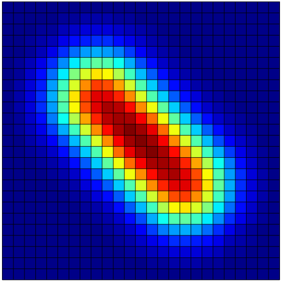
Covariance Functions

RBF Basis Functions

$$k(\mathbf{x}, \mathbf{x}') = \alpha \boldsymbol{\phi}(\mathbf{x})^\top \boldsymbol{\phi}(\mathbf{x}')$$

$$\phi_k(x) = \exp\left(-\frac{\left\|x - \mu_k\right\|_2^2}{\ell^2}\right)$$

$$\boldsymbol{\mu} = \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}$$



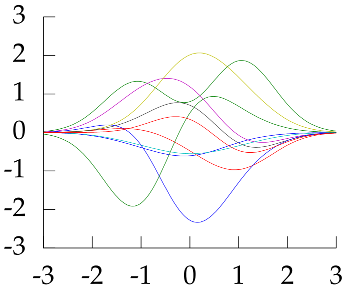
Covariance Functions

RBF Basis Functions

$$k(\mathbf{x}, \mathbf{x}') = \alpha \boldsymbol{\phi}(\mathbf{x})^\top \boldsymbol{\phi}(\mathbf{x}')$$

$$\phi_k(x) = \exp\left(-\frac{\left\|x - \mu_k\right\|_2^2}{\ell^2}\right)$$

$$\boldsymbol{\mu} = \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}$$



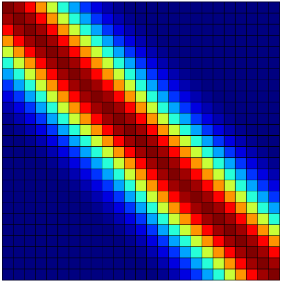
Covariance Functions

Where did this covariance matrix come from?

Exponentiated Quadratic Kernel Function (RBF, Squared Exponential, Gaussian)

$$k(\mathbf{x}, \mathbf{x}') = \alpha \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|_2^2}{2\ell^2}\right)$$

- Covariance matrix is built using the *inputs* to the function $\mathbf{x}$.
- For the example above it was based on Euclidean distance.
- The covariance function is also know as a kernel.



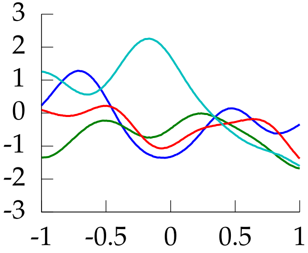
Covariance Functions

Where did this covariance matrix come from?

Exponentiated Quadratic Kernel Function (RBF, Squared Exponential, Gaussian)

$$k(\mathbf{x}, \mathbf{x}') = \alpha \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|_2^2}{2\ell^2}\right)$$

- Covariance matrix is built using the *inputs* to the function $\mathbf{x}$.
- For the example above it was based on Euclidean distance.
- The covariance function is also know as a kernel.



Gaussian Process Interpolation

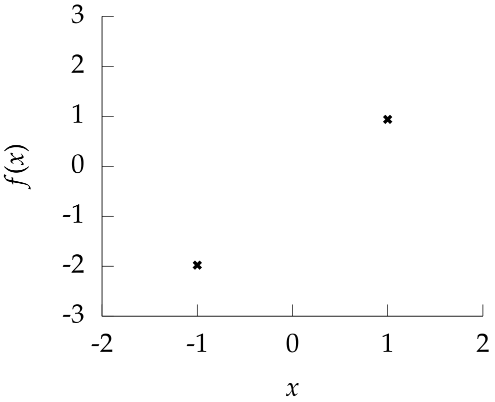


Figure: Real example: BACCO (see *e.g.* [3]). Interpolation through outputs from slow computer simulations (*e.g.* atmospheric carbon levels).

Gaussian Process Interpolation

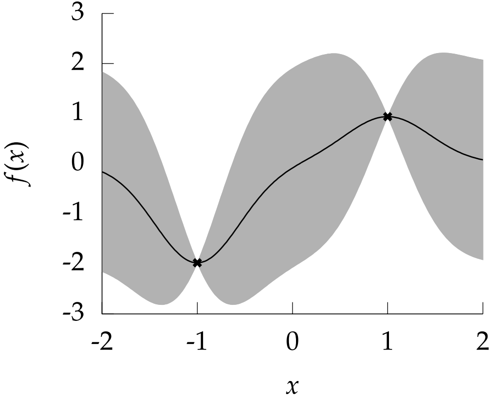


Figure: Real example: BACCO (see *e.g.* [3]). Interpolation through outputs from slow computer simulations (*e.g.* atmospheric carbon levels).

Gaussian Process Interpolation

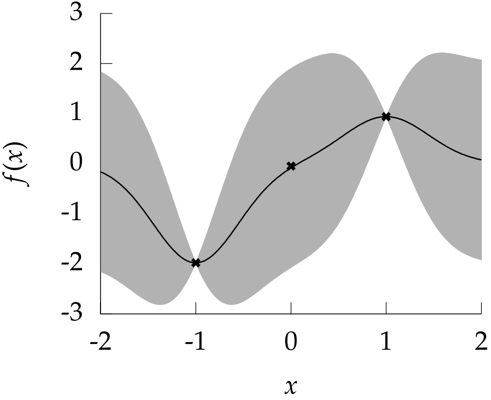


Figure: Real example: BACCO (see *e.g.* [3]). Interpolation through outputs from slow computer simulations (*e.g.* atmospheric carbon levels).

Gaussian Process Interpolation

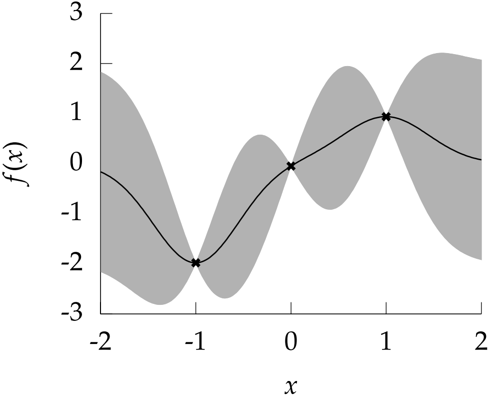


Figure: Real example: BACCO (see *e.g.* [3]). Interpolation through outputs from slow computer simulations (*e.g.* atmospheric carbon levels).

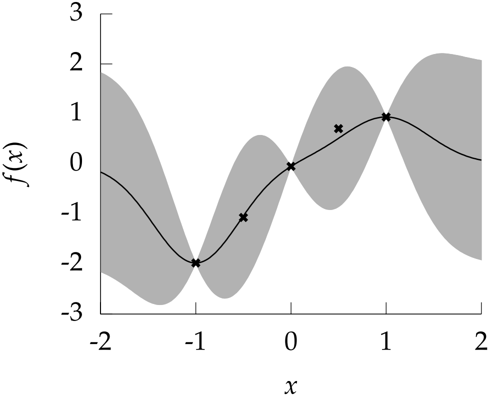


Figure: Real example: BACCO (see *e.g.* [3]). Interpolation through outputs from slow computer simulations (*e.g.* atmospheric carbon levels).

Gaussian Process Interpolation

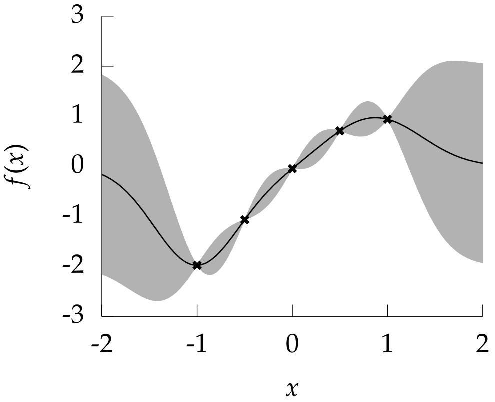


Figure: Real example: BACCO (see *e.g.* [3]). Interpolation through outputs from slow computer simulations (*e.g.* atmospheric carbon levels).

Gaussian Process Interpolation

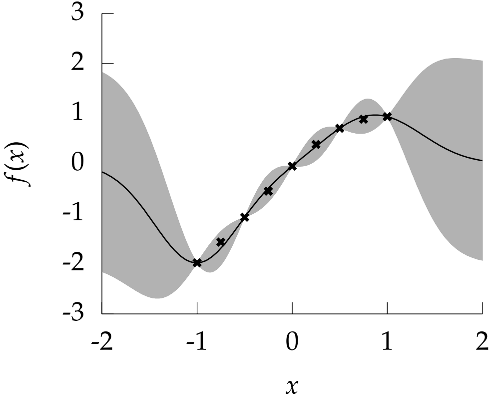


Figure: Real example: BACCO (see *e.g.* [3]). Interpolation through outputs from slow computer simulations (*e.g.* atmospheric carbon levels).

Gaussian Process Interpolation

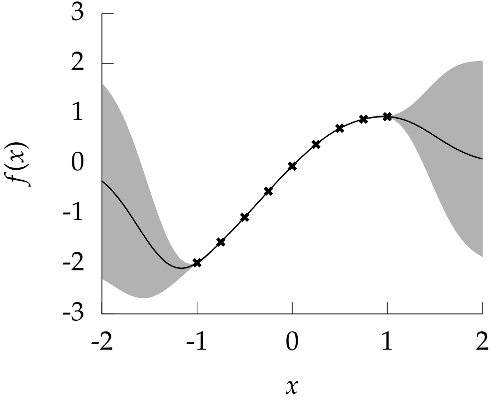


Figure: Real example: BACCO (see *e.g.* [3]). Interpolation through outputs from slow computer simulations (*e.g.* atmospheric carbon levels).

Gaussian Noise

- Gaussian noise model,

$$p\left(y_i|f_i\right) = \mathcal{N}\left(y_i|f_i, \sigma^2\right)$$

  where σ^2 is the variance of the noise.
- Equivalent to a covariance function of the form

$$k(\mathbf{x}_i, \mathbf{x}_j) = \delta_{i,j}\sigma^2$$

  where $\delta_{i,j}$ is the Kronecker delta function.
- Additive nature of Gaussians means we can simply add this term to existing covariance matrices.

Gaussian Process Regression

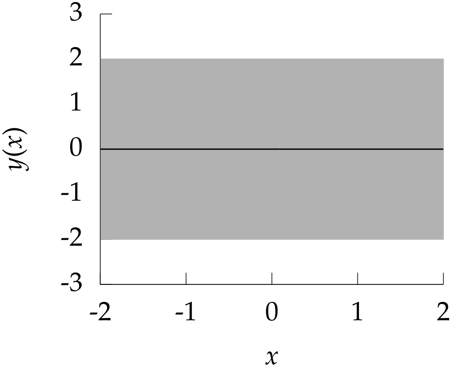


Figure: Examples include WiFi localization, C14 callibration curve.

Gaussian Process Regression

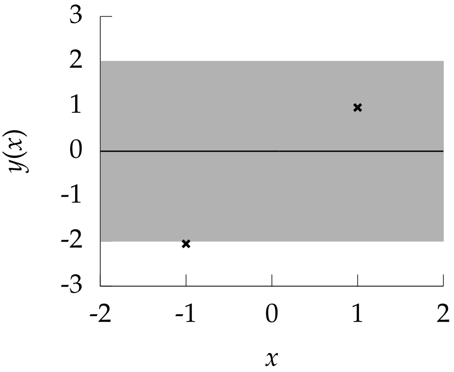


Figure: Examples include WiFi localization, C14 callibration curve.

Gaussian Process Regression

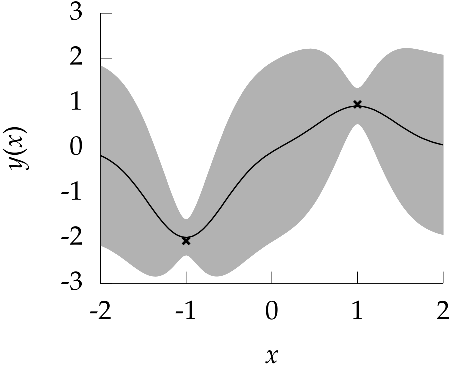


Figure: Examples include WiFi localization, C14 callibration curve.

Gaussian Process Regression

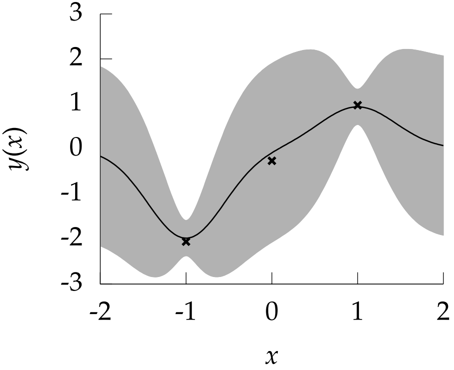


Figure: Examples include WiFi localization, C14 callibration curve.

Gaussian Process Regression

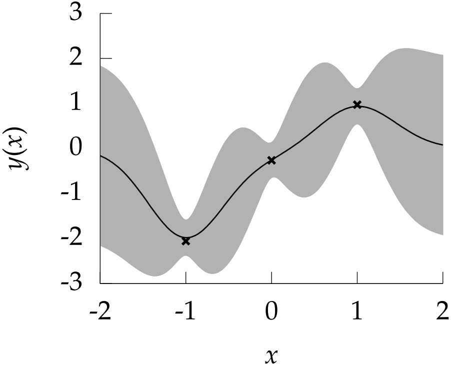


Figure: Examples include WiFi localization, C14 callibration curve.

Gaussian Process Regression

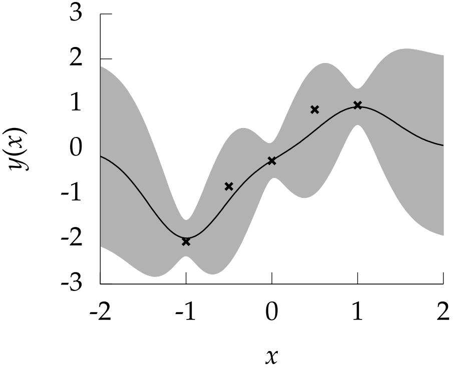


Figure: Examples include WiFi localization, C14 callibration curve.

Gaussian Process Regression

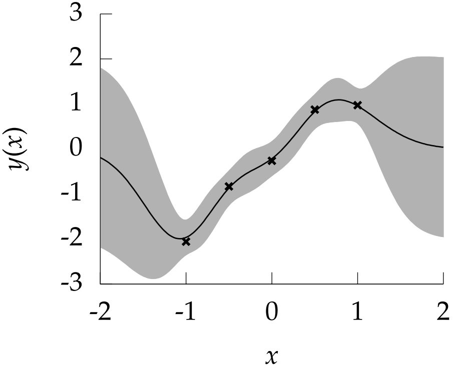


Figure: Examples include WiFi localization, C14 callibration curve.

Gaussian Process Regression

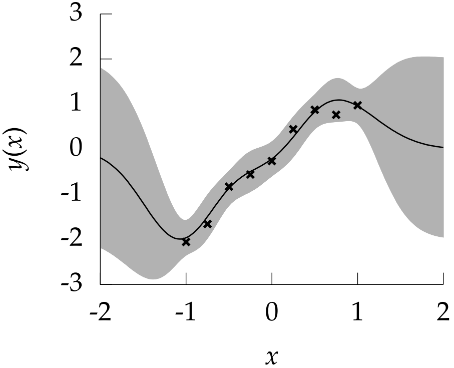


Figure: Examples include WiFi localization, C14 callibration curve.

Gaussian Process Regression

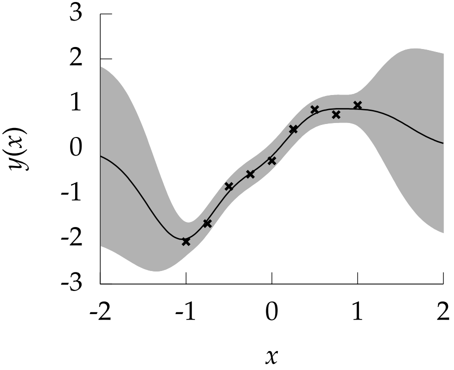


Figure: Examples include WiFi localization, C14 callibration curve.

- Learning with Gaussian processes allows to characterize the problem uncertainty:.
- We can choose a basis of functions o directly to select a kernel (equivalent, but better to choose the kernel: non-parametric models).
- Given a covariance (prior), how to select the right parameters?
- Back to optimization...

Learning Covariance Parameters

Can we determine covariance parameters from the data?

$$\mathcal{N}\left(\mathbf{y}|\mathbf{0},\mathbf{K}\right) = \frac{1}{(2\pi)^{\frac{n}{2}}|\mathbf{K}|^{\frac{1}{2}}}\exp\left(-\frac{\mathbf{y}^{\top}\mathbf{K}^{-1}\mathbf{y}}{2}\right)$$

The parameters are *inside* the covariance function (matrix).

$$k_{i,j} = k(\mathbf{x}_i, \mathbf{x}_j; \boldsymbol{\theta})$$

Learning Covariance Parameters

Can we determine covariance parameters from the data?

$$\mathcal{N}\left(\mathbf{y}|\mathbf{0},\mathbf{K}\right)=\frac{1}{(2\pi)^{\frac{n}{2}}|\mathbf{K}|^{\frac{1}{2}}}\exp\left(-\frac{\mathbf{y}^{\top}\mathbf{K}^{-1}\mathbf{y}}{2}\right)$$

The parameters are *inside* the covariance function (matrix).

$$k_{i,j}=k(\mathbf{x}_i,\mathbf{x}_j;\boldsymbol{\theta})$$

Learning Covariance Parameters

Can we determine covariance parameters from the data?

$$\log \mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{K}) = -\frac{1}{2}\log|\mathbf{K}| - \frac{\mathbf{y}^\top \mathbf{K}^{-1}\mathbf{y}}{2} - \frac{n}{2}\log 2\pi$$

The parameters are *inside* the covariance function (matrix).

$$k_{i,j} = k(\mathbf{x}_i, \mathbf{x}_j; \boldsymbol{\theta})$$

Learning Covariance Parameters

Can we determine covariance parameters from the data?

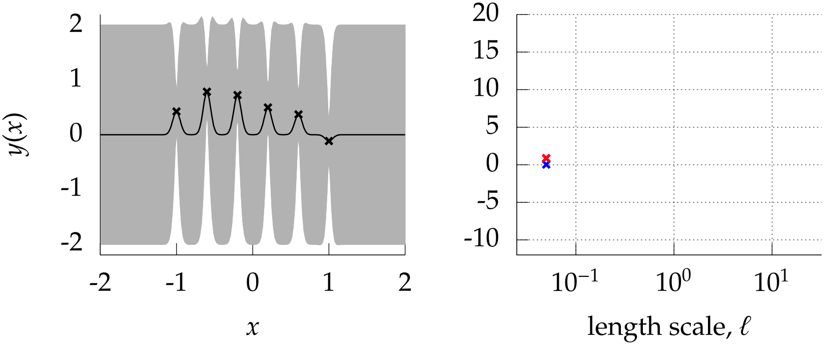
$$E(\boldsymbol{\theta}) = \frac{1}{2}\log|\mathbf{K}| + \frac{\mathbf{y}^\top \mathbf{K}^{-1}\mathbf{y}}{2}$$

The parameters are *inside* the covariance function (matrix).

$$k_{i,j} = k(\mathbf{x}_i, \mathbf{x}_j; \boldsymbol{\theta})$$

Learning Covariance Parameters

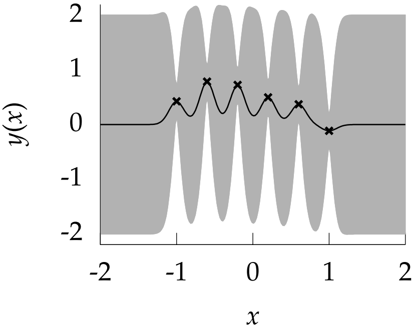
Can we determine length scales and noise levels from the data?

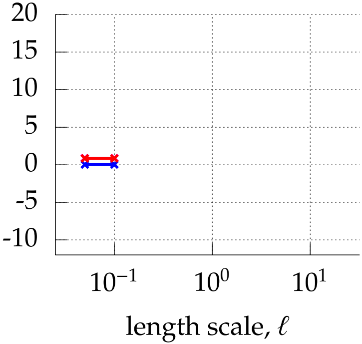


$$E(\boldsymbol{\theta}) = \frac{1}{2}\log|\mathbf{K}| + \frac{\mathbf{y}^\top\mathbf{K}^{-1}\mathbf{y}}{2}$$

Learning Covariance Parameters

Can we determine length scales and noise levels from the data?

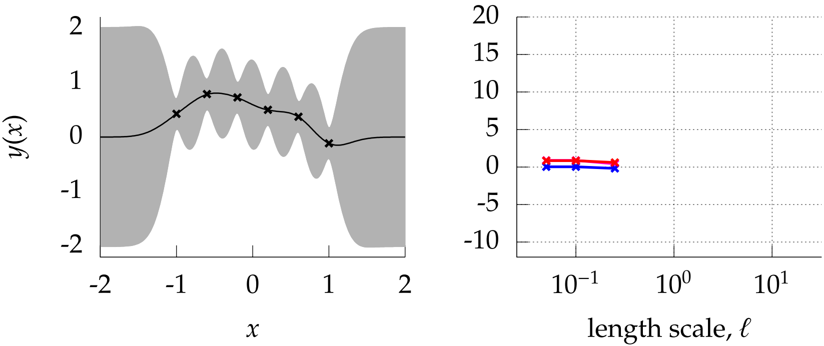




$$E(\boldsymbol{\theta}) = \frac{1}{2}\log|\mathbf{K}| + \frac{\mathbf{y}^{\top}\mathbf{K}^{-1}\mathbf{y}}{2}$$

Learning Covariance Parameters

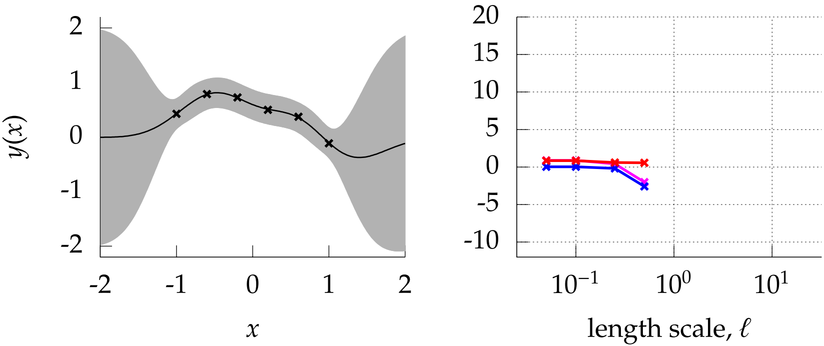
Can we determine length scales and noise levels from the data?



$$E(\boldsymbol{\theta}) = \frac{1}{2}\log|\mathbf{K}| + \frac{\mathbf{y}^{\top}\mathbf{K}^{-1}\mathbf{y}}{2}$$

Learning Covariance Parameters

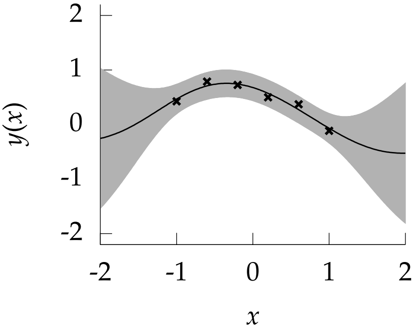
Can we determine length scales and noise levels from the data?

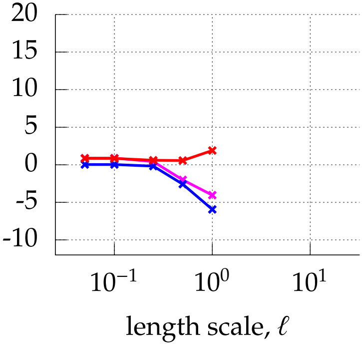


$$E(\boldsymbol{\theta}) = \frac{1}{2}\log|\mathbf{K}| + \frac{\mathbf{y}^{\top}\mathbf{K}^{-1}\mathbf{y}}{2}$$

Learning Covariance Parameters

Can we determine length scales and noise levels from the data?

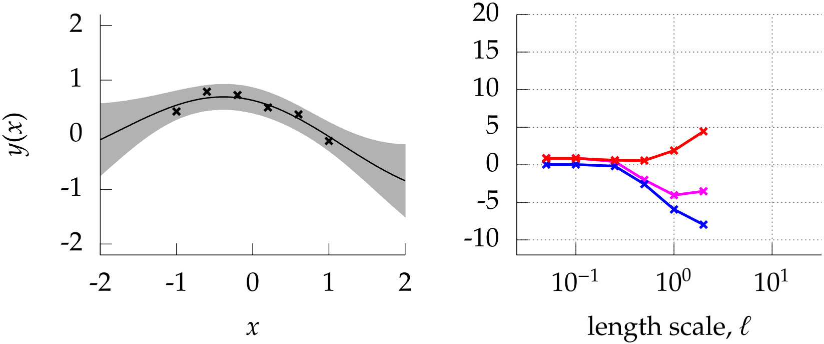




$$E(\boldsymbol{\theta}) = \frac{1}{2}\log|\mathbf{K}| + \frac{\mathbf{y}^\top\mathbf{K}^{-1}\mathbf{y}}{2}$$

Learning Covariance Parameters

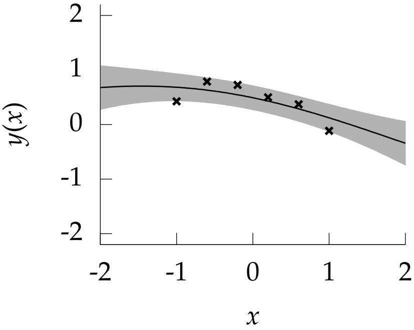
Can we determine length scales and noise levels from the data?

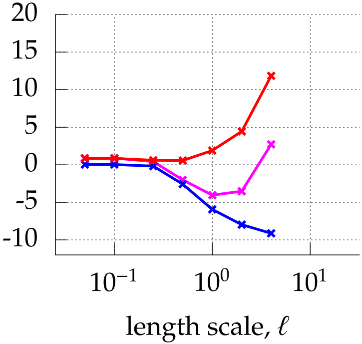


$$E(\boldsymbol{\theta}) = \frac{1}{2} \log |\mathbf{K}| + \frac{\mathbf{y}^{\top} \mathbf{K}^{-1} \mathbf{y}}{2}$$

Learning Covariance Parameters

Can we determine length scales and noise levels from the data?

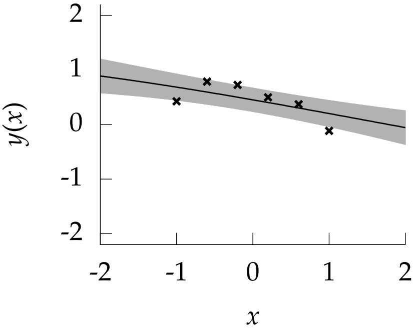


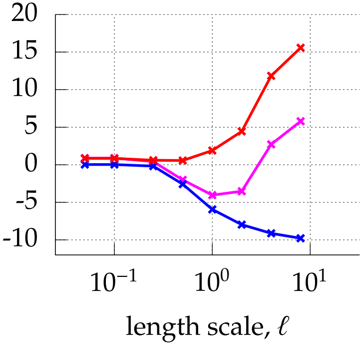


$$E(\boldsymbol{\theta}) = \frac{1}{2}\log|\mathbf{K}| + \frac{\mathbf{y}^\top\mathbf{K}^{-1}\mathbf{y}}{2}$$

Learning Covariance Parameters

Can we determine length scales and noise levels from the data?

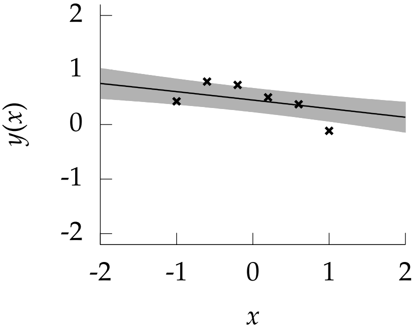


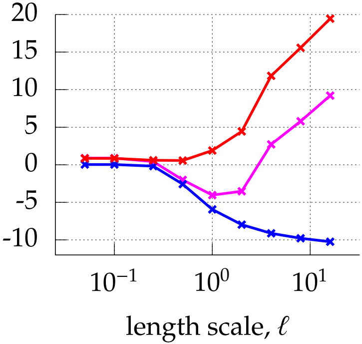


$$E(\boldsymbol{\theta}) = \frac{1}{2}\log|\mathbf{K}| + \frac{\mathbf{y}^\top\mathbf{K}^{-1}\mathbf{y}}{2}$$

Learning Covariance Parameters

Can we determine length scales and noise levels from the data?





$$E(\boldsymbol{\theta}) = \frac{1}{2}\log|\mathbf{K}| + \frac{\mathbf{y}^{\top}\mathbf{K}^{-1}\mathbf{y}}{2}$$

Limitations of Gaussian Processes

- Inference is $O(n^3)$ due to matrix inverse (in practice use Cholesky).
- Gaussian processes don't deal well with discontinuities (financial crises, phosphorylation, collisions, edges in images).
- Widely used exponentiated quadratic covariance (RBF) can be too smooth in practice (but there are many alternatives!!).

Conclusions

- Machine learning has focussed on prediction.
- Two main approaches: optimize objective, or model probabilistically.
- Both approaches require to optimize parameters.
- Gaussian processes: fundamental models to deal with uncertainty in complex scenarios.

Tomorow

- Global optimization.
- Parameter tuning in Machine Learning as a global optimization problem.
- Can we automate the parameter choice of Machine Learning algorithms?
- Yes! Bayesian Optimization.

References I

[1] Carl Friedrich Gauss. *Theoria motus corporum coelestium*. Perthes et Besser, Hamburg.

[2] Pierre Simon Laplace. Mémoire sur la probabilité des causes par les évènemens. In *Mémoires de mathèmatique et de physique, presentés à lAcadémie Royale des Sciences, par divers savans, & lù dans ses assemblées 6*, pages 621–656, 1774. Translated in [5].

[3] Jeremey Oakley and Anthony O'Hagan. Bayesian inference for the uncertainty distribution of computer model outputs. *Biometrika*, 89(4):769–784, 2002.

[4] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, MA, 2006.

[5] Stephen M. Stigler. Laplace's 1774 memoir on inverse probability. *Statistical Science*, 1:359–378, 1986.