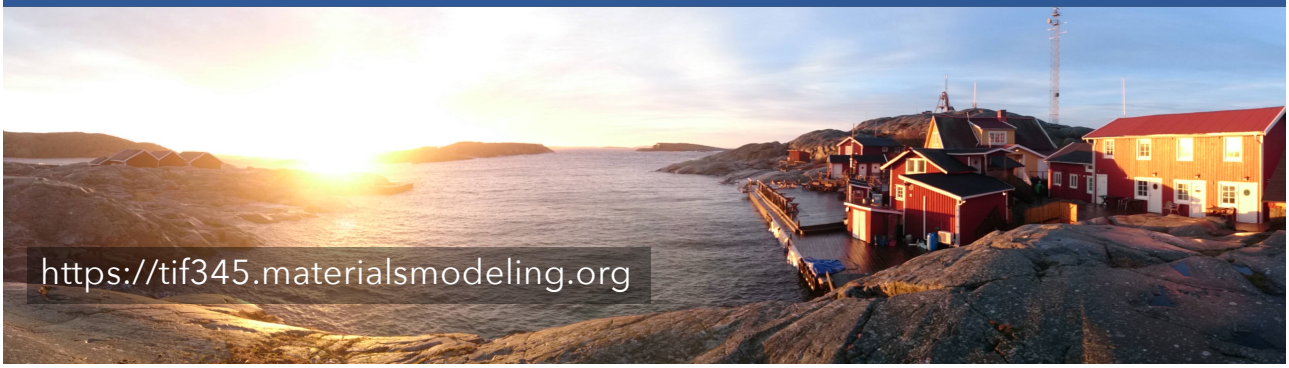
TIF345 / FYM345
Advanced Simulation and Machine Learning

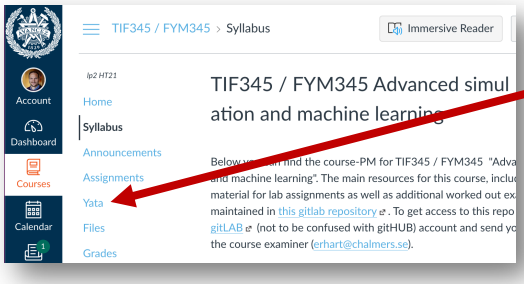
Toward applications and Advanced Regression

Paul Erhart (examiner)
Andreas Ekström
Bernhard Mehlig
Arkady Gonoskov



Course Resources

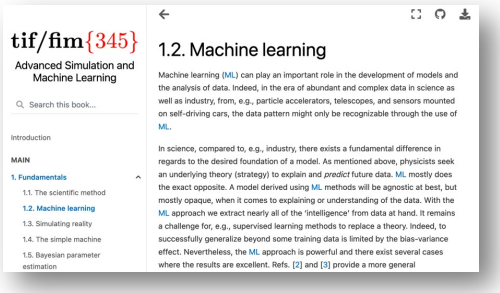
Course homepage canvas



Yata (forum)

hypothes.is

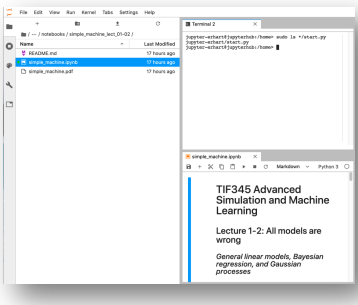
Lecture notes



Code GitLab

tif/fim {345}

For labs jupyterhub



B. Mehlig, "Machine Learning with Neural Networks: An Introduction for Scientists and Engineers"

CHALMERS UNIVERSITY OF TECHNOLOGY

Plan for the next two weeks

tif/fim{345}

Advanced Simulation and Machine Learning

1-2 Foundations: Linear models, Gaussian processes

3-4 Applications: Linear models, Regression, (Gaussian processes)

5-7 Neural networks and other ML techniques

L1: Advanced regression

L2: Applications

→ P2a: Regression with cluster expansions

L3: Sensitivity analysis

L4: Global optimization

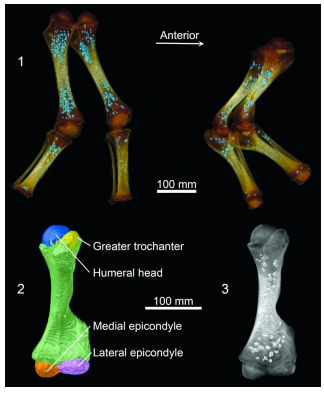
→ P2b: GPs and Bayesian optimization

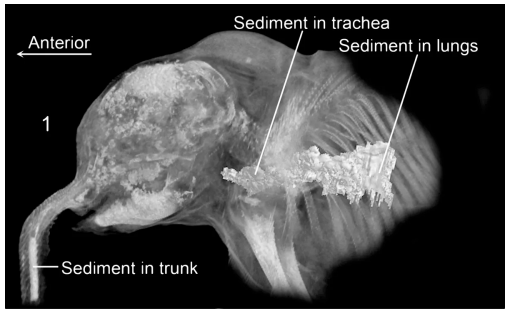
Demos

- 5. Advanced regression
  - 5.1. Ridge regression and beyond
  - 5.2. Robust regression
  - 5.3. Error correlation
  - 5.4. Additional considerations
  - 5.5. A brief detour into frequentist statistics
  - 5.6. Feature selection and sparse models
  - 5.7. Key take-aways
- 6. Toward applications
  - 6.1. Alloy cluster expansions
  - 6.2. Phonons and force constants
  - 6.3. Interatomic potentials

Applications: Tomography

X-ray tomography

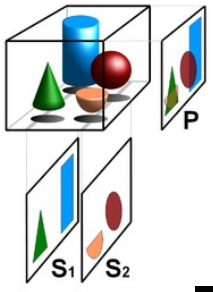




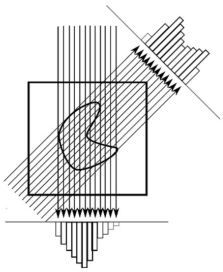
https://www.livescience.com/16927-baby-mammoth-ct-scans.html

Applications: Tomography

Working principle
generate 3D image
from series of 2D data



Radon transform (2D)



$$\mathcal{R}\big[f(x,y)\big](H,\phi) = \int_{-\infty}^{\infty} f\big(H(\sin\phi,\cos\phi) + s(\cos\phi,\sin\phi)\big)\,ds$$

Function on x-y plane, e.g., density

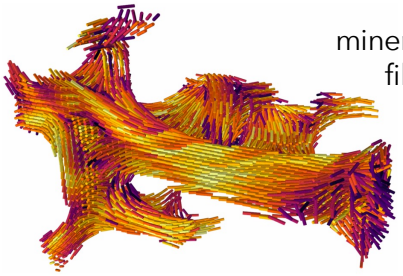
→ Can be cast as **generalized linear model**

What else could *f* be?

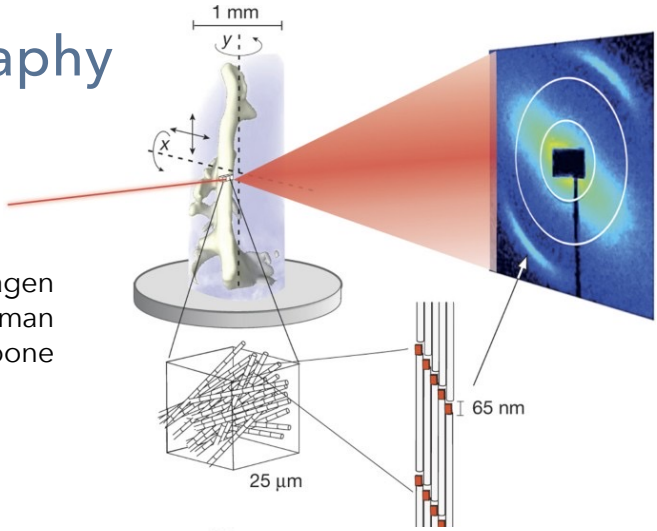
Pictures from Wikipedia,
https://imgflip.com/gif/29e1g4

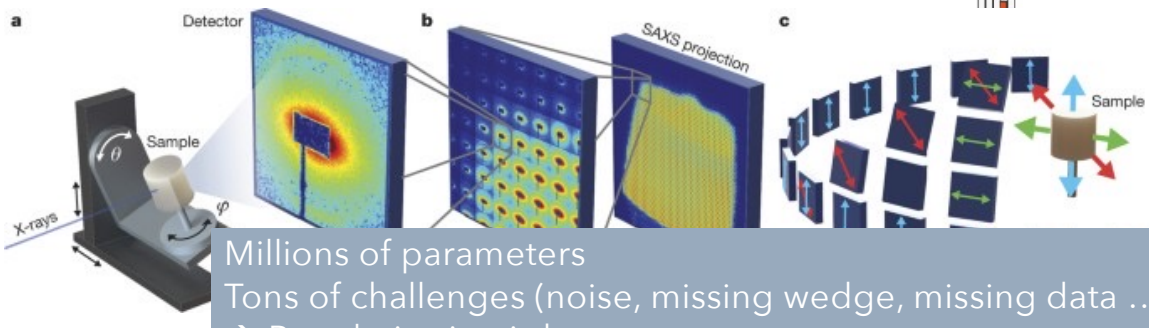
Applications: Tomography

From scalar to tensor tomography
→ Orientation distribution



mineralized collagen
fibrils in a human
trabecula bone





Millions of parameters
Tons of challenges (noise, missing wedge, missing data …)
→ Regularization is key

Liebi *et al.*, Nature **527** (2015), Schaff *et al.*, *ibid.*

Applications: Atomic scale models

Task: Map positions to potential energy surface (or other properties)

- Interatomic potentials $\{\boldsymbol{R}_i\} \rightarrow E \qquad \boldsymbol{f}_i = -dE/d\boldsymbol{R}_i$

Generalized linear models

- Force constant expansions $\{\boldsymbol{u}_i\} \rightarrow E \quad \boldsymbol{u}_i = \boldsymbol{R}_i - \boldsymbol{R}_i^{(0)}$

  $\boldsymbol{f}_i = -\boldsymbol{\Phi}_{ij}\boldsymbol{u}_j - \boldsymbol{\Phi}_{ijk}\boldsymbol{u}_j\boldsymbol{u}_k \ldots$

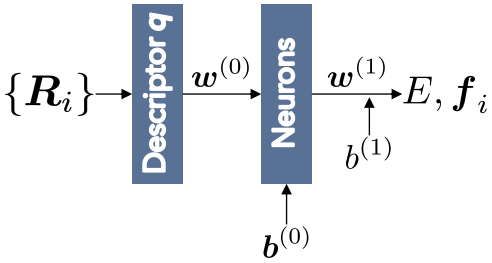
  Displacements
- Cluster expansions $\{\sigma_i\} \rightarrow E$

  Occupation vector

Hundreds to tens of thousands parameters
→ Regularization is key

Applications: Atomic scale models

Example: Neuroevolution potential (NEP) models



$$E = \sum_i E_i$$

$$E_i = \sum_\mu^{\text{neu}} w_\mu^{(1)} \tanh\left(\sum_\nu^{\text{des}} w_{\mu\nu}^{(0)} q_\nu^{(i)} - b_\mu^{(0)}\right) - b^{(1)}$$

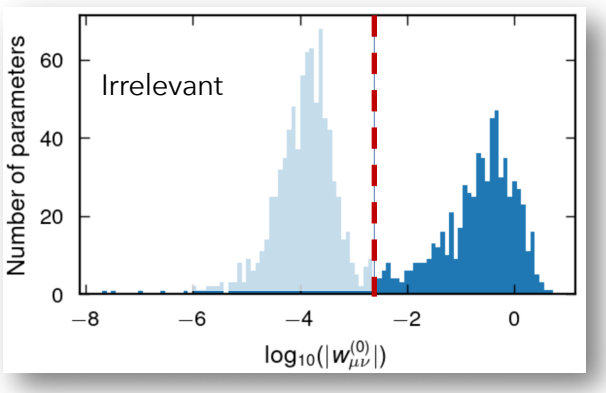
Optimization

separable natural evolution strategy

Loss function

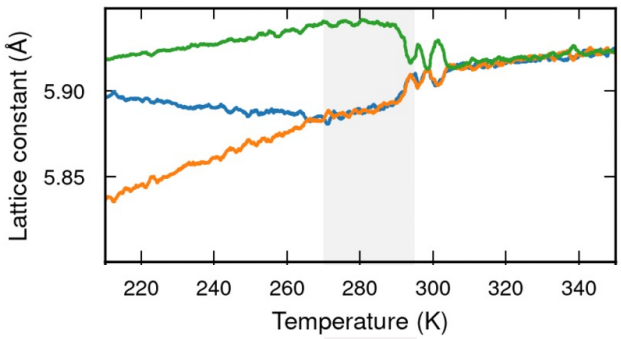
$$\begin{aligned}\mathcal{L}(\boldsymbol{w}) = {} & \lambda_{\text{energy}} \cdot \text{RMSE(energy)} \\ & + \lambda_{\text{forces}} \cdot \text{RMSE(forces)} \\ & + \lambda_{\text{virial}} \cdot \text{RMSE(virial)} \\ & + \lambda_1 \cdot \|\boldsymbol{w}\|_1 + \lambda_2 \cdot \|\boldsymbol{w}\|_2\end{aligned}$$

Regularization
→ stability
→ data efficiency



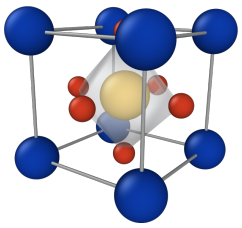
Applications: Atomic scale models

$CsPbBr_3$



Orthorhombic *Pnma*

$a^-a^-c^+$

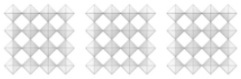


Tetragonal *P4/mbm*

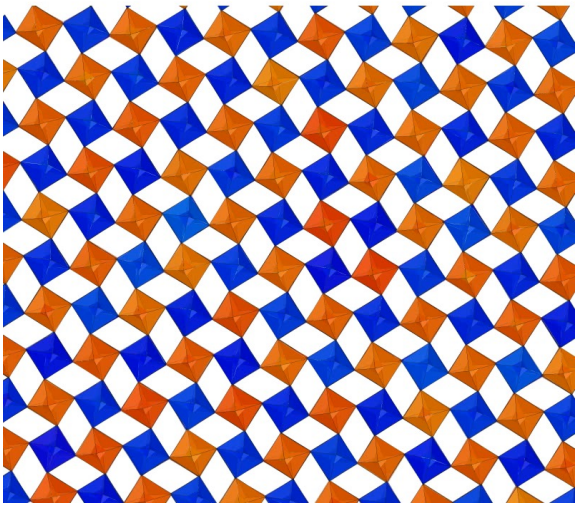
$a^0a^0c^+$

Cubic *Pm$\bar{3}$m*

$a^0a^0a^0$



20 K 0.5 ns



~64,000 atoms

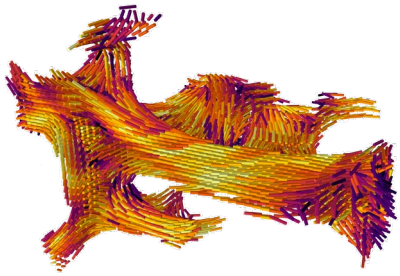
CHALMERS
UNIVERSITY OF TECHNOLOGY

Practical challenges

(Very) large **data** sets
Incomplete data sets

(Very) large **parameter** spaces

Errors / **noise**



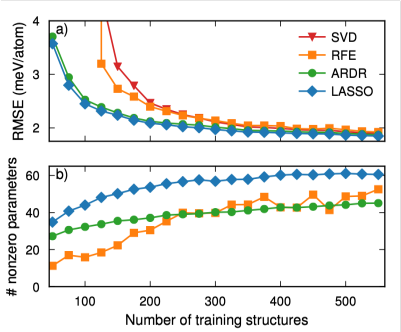
Potentially functional forms
that are numerically problematic ("sloppy models")

Consequences

- Direct MCMC often impractical
- No general closed solutions for non-linear problems

→ Model design and problem formulation important

Follow-up questions (1)



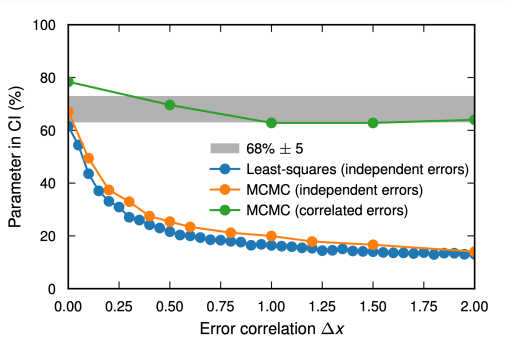
- How do we find out which parameters matter?
  → sparse models via regularization

- How do we do this efficiently while accounting meaningfully for prior information?
  → loss function

$$\mathcal{L} = \boldsymbol{\varepsilon}^T\boldsymbol{\varepsilon} + \lambda_1\|\boldsymbol{\theta}\|_1 + \lambda_2\|\boldsymbol{\theta}\|_2^2 + \lambda_3\|\nabla^2\boldsymbol{\theta}\|_2^2 + \ldots$$

- How we handle error correlation?
  → "Sigma"

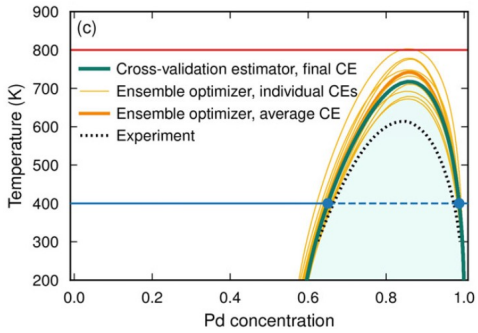
$$\mathcal{L} = \boldsymbol{\varepsilon}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\varepsilon} + \ldots$$



CHALMERS
UNIVERSITY OF TECHNOLOGY

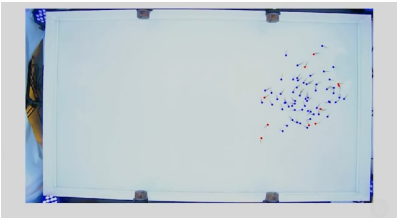
Follow-up questions (2)

- How can we quantify model uncertainty?
  → Sensitivity analysis and error propagation
  (e.g., bagging, committee/ensemble models, Mahalanobis norm),



- How do we acquire data?
  → active learning

- How do find solutions in complex and often high-dimensional parameter spaces?
  → global optimization

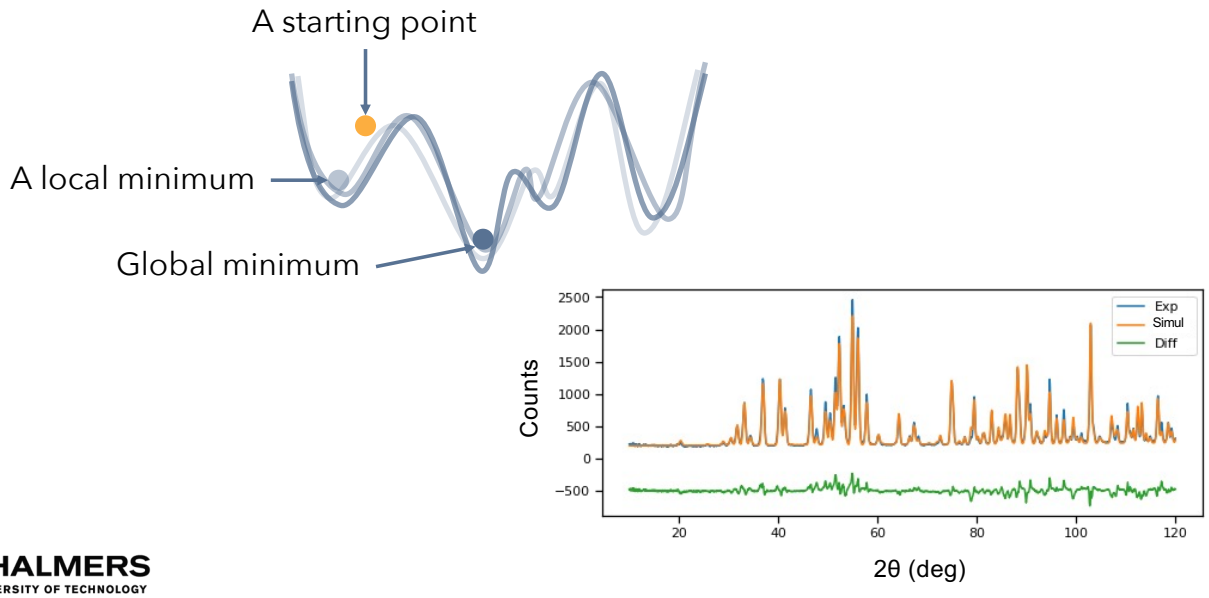


CHALMERS
UNIVERSITY OF TECHNOLOGY

Reconsider: What are the benefits of a Bayesian perspective?

Problem statement

1.	What do we optimize?	→ linear vs non-linear models
2.	How do we optimize?	→ local vs global optimization
3.	What about errors?	→ error correlation
4.	How do we convey prior insight?	→ priors and covariances



A motivating example

$$y = a_1 \exp\left(-\frac{x}{t_1}\right) + a_2 \exp\left(-\frac{x - 0.1}{t_2}\right)$$

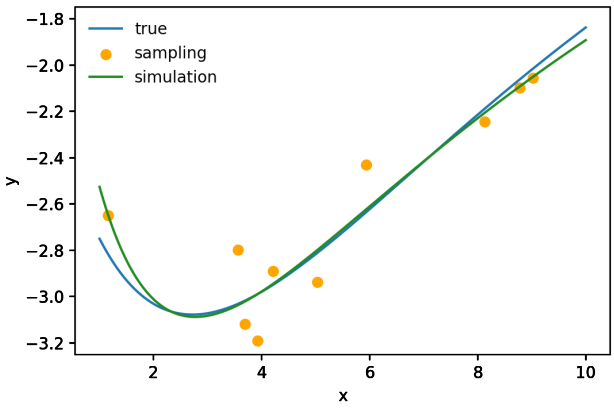
```
import numpy as np
from scipy.optimize import curve_fit
```

Truth

4 parameters

```
def my_func(x, a1, a2, t1, t2):
    return a1 * np.exp(-x / t1) \
         + a2 * np.exp(- (x - 0.1) / t2)
```

```
params_true = (3.0, -5.0, 2.0, 10.0)
```



Sampling

10 data points

```
xs_sampled = 9 * np.random.random_sample(size=10) + 1
ys_sampled = my_func(xs_sampled, *params_true)
ys_sampled += np.random.normal(scale=0.1, size=len(xs_sampled))
```

add noise

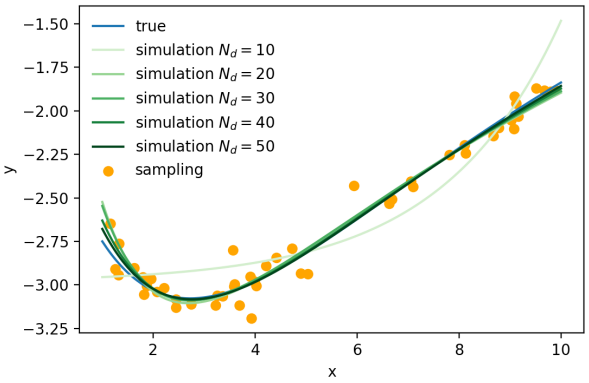
Simulation/inference/training

```
params_simulated, params_cov = curve_fit(my_func, xs_sampled, ys_sampled)
```

Least-squares

A motivating example

- Adding more data does not necessarily imply convergence
- How can uncertainty be assessed?



Covariance

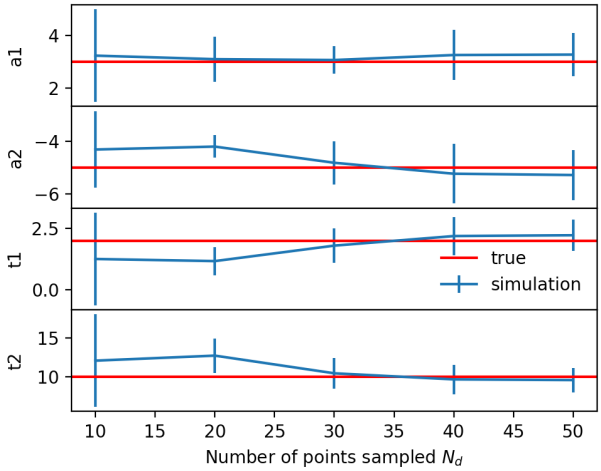
```
params_simulated, params_cov = curve_fit(
```

Parameter estimates with uncertainties

```
params_cov
```

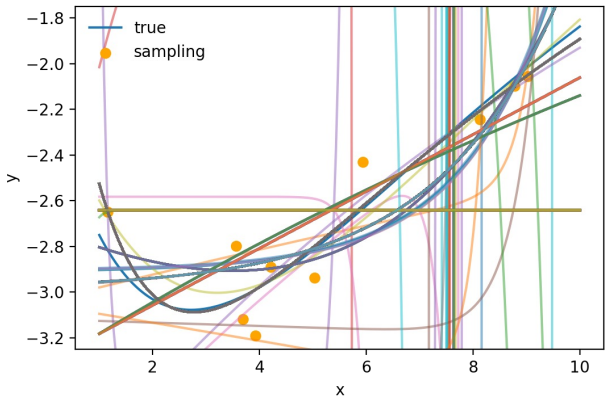
```
array([[ 0.059, -0.074,  0.052, -0.204],
       [-0.074,  0.176, -0.159,  0.461],
       [ 0.052, -0.159,  0.152, -0.408],
       [-0.204,  0.461, -0.408,  1.222]])
```

```
params_errors = np.sqrt(np.diag(params_cov))
```

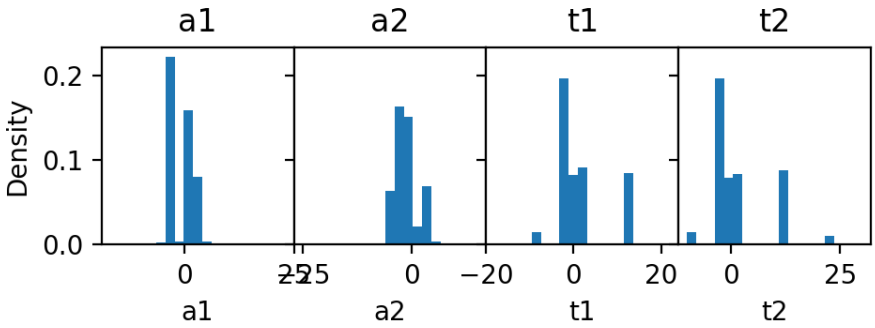


A motivating example

Results can be very sensitive to the initial conditions



Distribution of "optimal" parameters

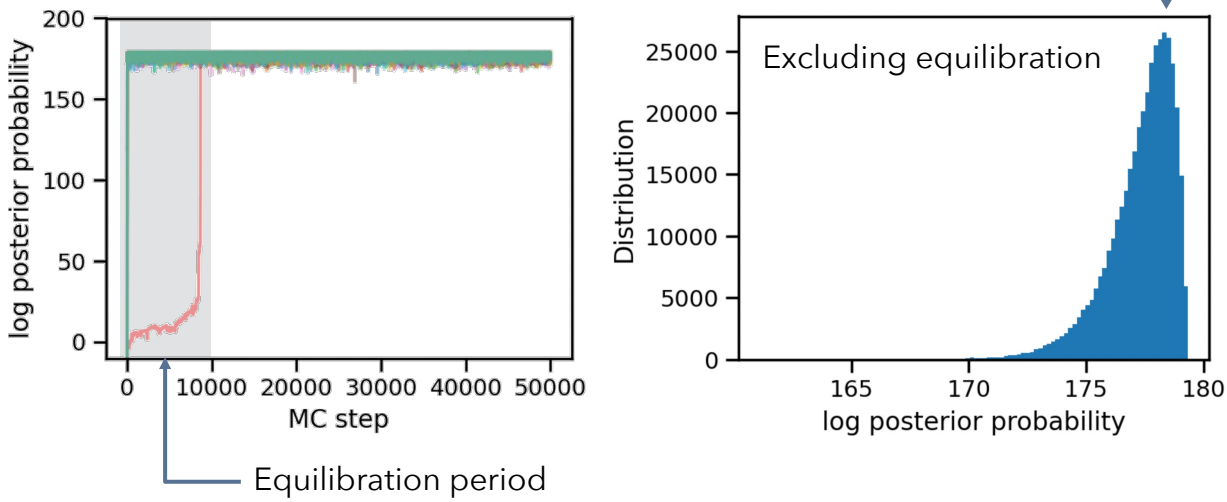


Monte Carlo sampling

- Sampling of posterior via Markov chain-Monte Carlo simulations

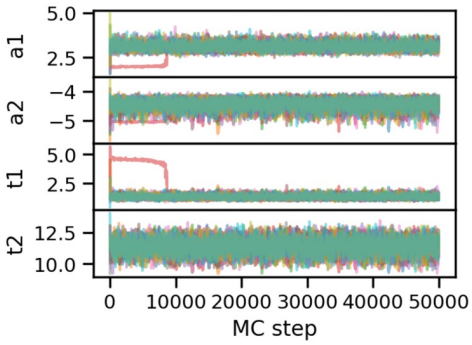
Not a Gaussian!

log posterior probability $\ln p(\boldsymbol{\theta}, \sigma^2 | \boldsymbol{\mathcal{D}})$

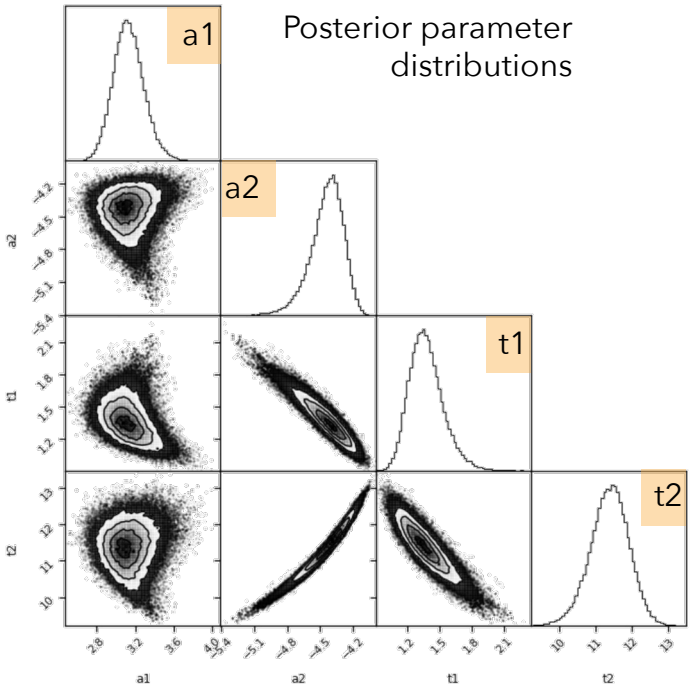


Posterior parameter estimates

MCMC trajectories

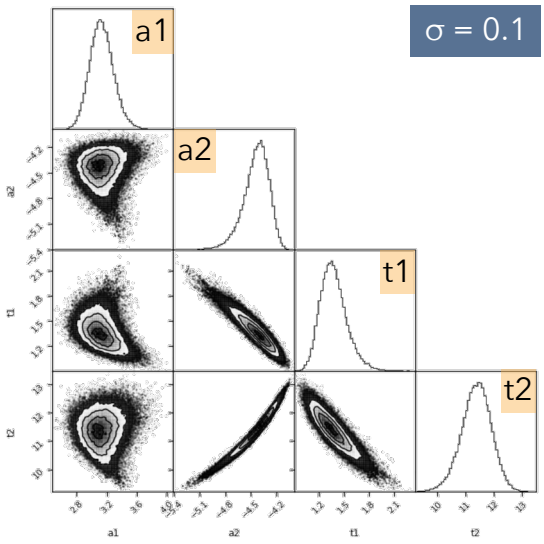


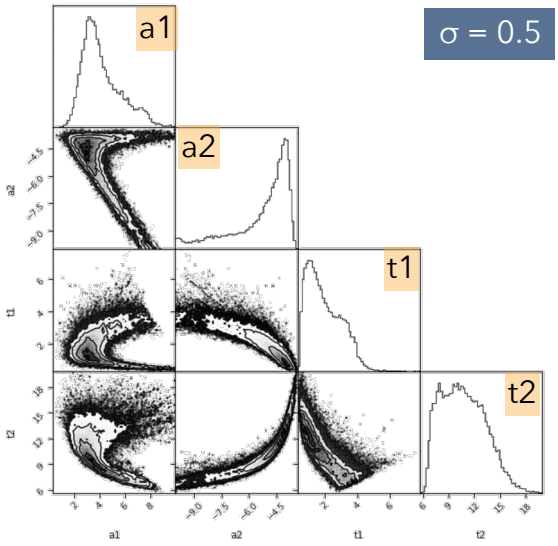
Corner plot



How do error estimates in standard least-squares (`curve_fit`) work?

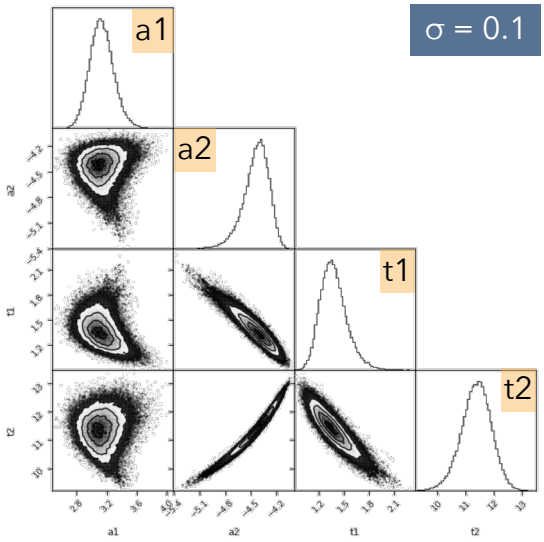
Posterior parameter estimates

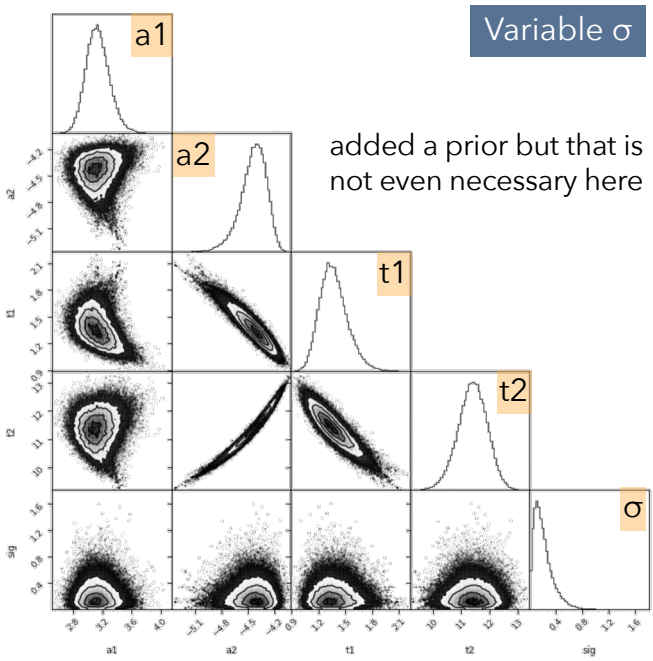




What is the "correct" σ? → Sample it!

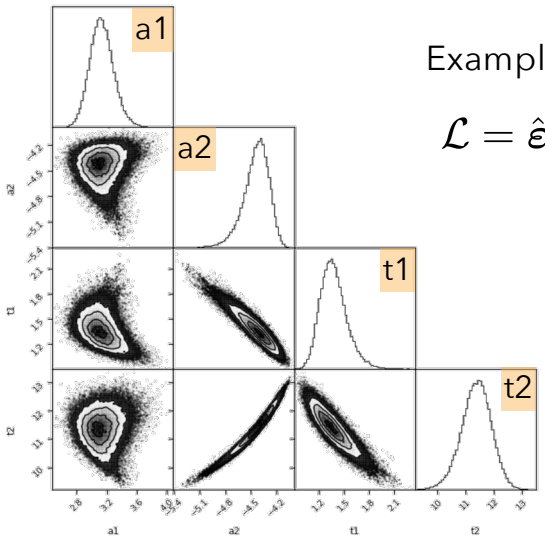
Posterior parameter estimates





added a prior but that is not even necessary here

What do we gain from a Bayesian perspective?



Example for a simple loss function

$$\mathcal{L} = \hat{\varepsilon}^T \boldsymbol{\Sigma}^{-1} \hat{\varepsilon} + \boldsymbol{\theta}^T \boldsymbol{\Lambda}^{-1} \boldsymbol{\theta}$$

Model information

Error information

- Deeper understanding of correlations
- A probabilistic view that we can apply for interpretation

$\Lambda = \lambda^2 1$

Ridge regression and beyond: Parameter specific priors

$\Lambda = \mathrm{diag}$

Ridge and Bayesian ridge regression

Ridge regression $p(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}|0, \tau^2)$

Bayesian ridge regression $p(\mathcal{D}; M(\boldsymbol{\theta}), \alpha) = \mathcal{N}(\mathcal{D}|M(\boldsymbol{\theta}), \alpha)$

$p(\boldsymbol{\theta}|\lambda) = \mathcal{N}(\boldsymbol{\theta}|0, \lambda^{-1}\boldsymbol{I})$

α and λ from Gamma distributions estimated jointly during fitting by maximizing the log marginal likelihood (LML)

Recall

$$\mathcal{L} = \hat{\varepsilon}^T \boldsymbol{\Sigma}^{-1} \hat{\varepsilon} + \boldsymbol{\theta}^T \boldsymbol{\Lambda}^{-1} \boldsymbol{\theta}$$

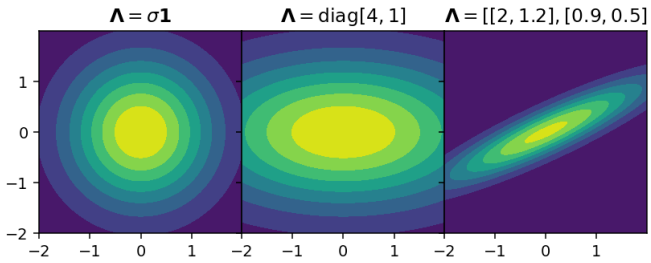
Error information

Model information

Automatic relevance detection (ARD) regression

$$p(\boldsymbol{\theta}|\boldsymbol{\Lambda}) = \mathcal{N}(\boldsymbol{\theta}|0, \boldsymbol{\Lambda}^{-1})$$
$$\text{diag}\boldsymbol{\Lambda} = \left[\lambda_1, \lambda_2 \ldots \lambda_{N_p}\right]$$

Now: Gamma distribution for <u>each</u> parameter

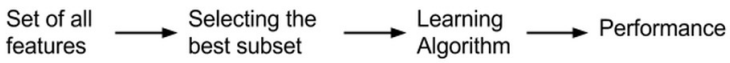


Followed by a pruning step where parameters for which the statistical significance is below a threshold are set to zero
→ Feature selection (next)

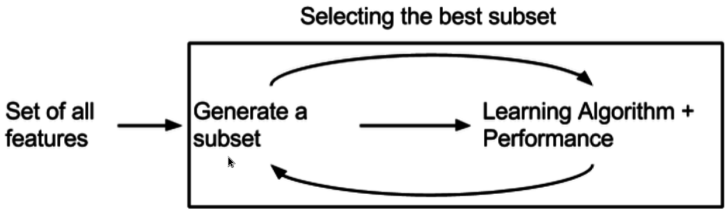
Feature selection: Making models sparse

(Automated) feature selection

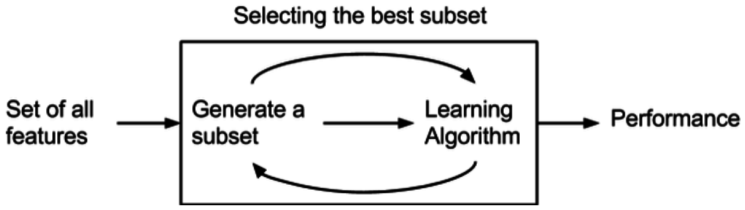
Filter-type



Embedded-type



Wrapper-type (RFE, genetic algorithms, ant colony optimization, …)



Figures from wiki commons and doi:10.1109/EMBC.2016.7591621

(Automated) feature selection

"Embedded"-type
Algorithms that "take advantage of their own variable selection"

Automatic relevance detection (ARD) regression

$$p(\boldsymbol{\theta}|\boldsymbol{\Lambda}) = \mathcal{N}(\boldsymbol{\theta}|0, \boldsymbol{\Lambda}^{-1})$$
$$\mathrm{diag}\boldsymbol{\Lambda} = \left[\lambda_1, \lambda_2 \ldots \lambda_{N_p}\right]$$

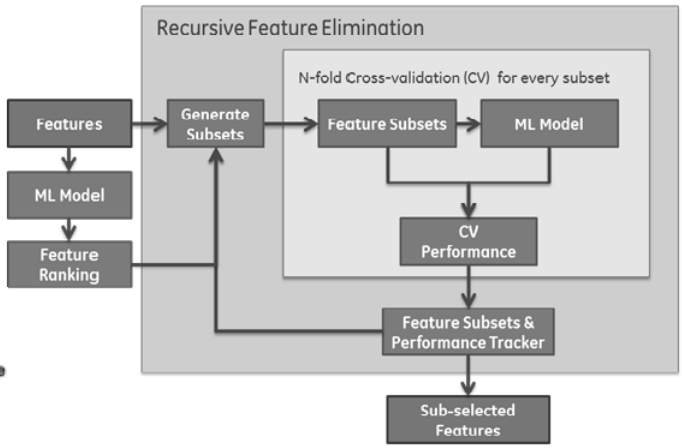
Followed by pruning step

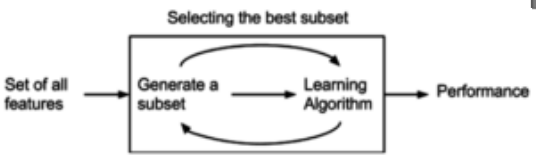
$$\hat{\theta}_i = \begin{cases} \theta_i & \text{if} \quad \lambda_i < \lambda_{\text{threshold}} \\ 0 & \text{else} \end{cases}$$

Principle can be similarly applied to LASSO

(Automated) feature selection

Wrapper-type
Algorithms that wrap around another regression technique
→ Example: Recursive feature elimination (RFE)





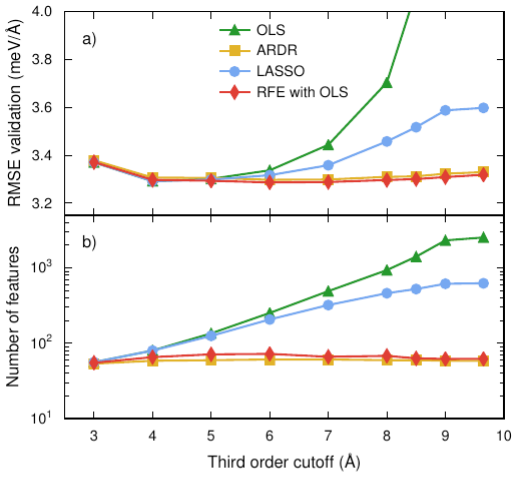

```python
from sklearn.feature_selection import RFE, RFECV
from sklearn.linear_model import LinearRegression

X, E = sc.get_fit_data()
est = LinearRegression(…)
rfe = RFE(estimator=est, n_features_to_select=…)
rfe.fit(X, E)
```

hyperparameter

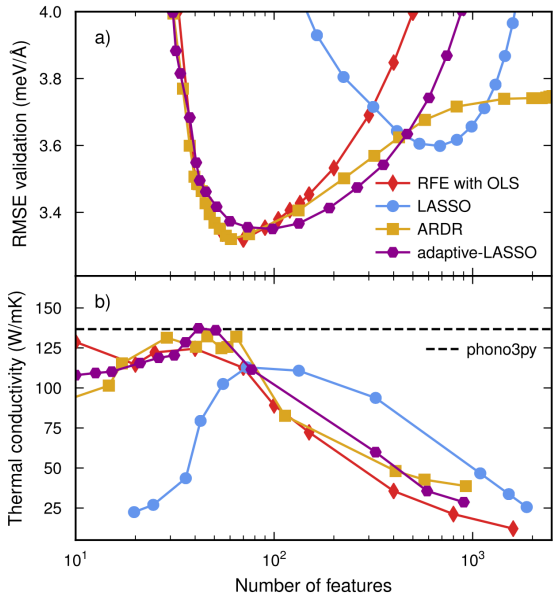
Figures from wiki commons and doi:10.1109/EMBC.2016.7591621

Ex: Thermal conductivity from force constants



Sparsity often implies "more physical" models

Often improves stability



Dealing with outliers: Robust regression

Dealing with non-normal iid errors

OLS, ridge, LASSO etc:
errors given by normal distribution (and iid)

$$p(\mathcal{D}; M(\boldsymbol{\theta}), \alpha) = \mathcal{N}(\mathcal{D}|M(\boldsymbol{\theta}), \alpha)$$

What happens when you have outliers or "heavy tails"?
→ notebooks 1 and 2

Robust regression:
assume heavy-tailed distribution

$$p(\mathcal{D}; M(\boldsymbol{\theta}), b) = \exp\left(-\frac{1}{b}|\mathcal{D} - M(\boldsymbol{\theta})|\right)$$ e.g., Laplace

The Huber regressor

Robust regression:
assume heavy-tailed distribution

$$p(\mathcal{D}; M(\boldsymbol{\theta}), b) = \exp\left(-\frac{1}{b}|\mathcal{D} - M(\boldsymbol{\theta})|\right)$$ e.g., Laplace

Problem: Non-differentiable loss function

$$\mathcal{L} = \|\mathcal{D} - M(\boldsymbol{\theta})\|_1 + \dots$$

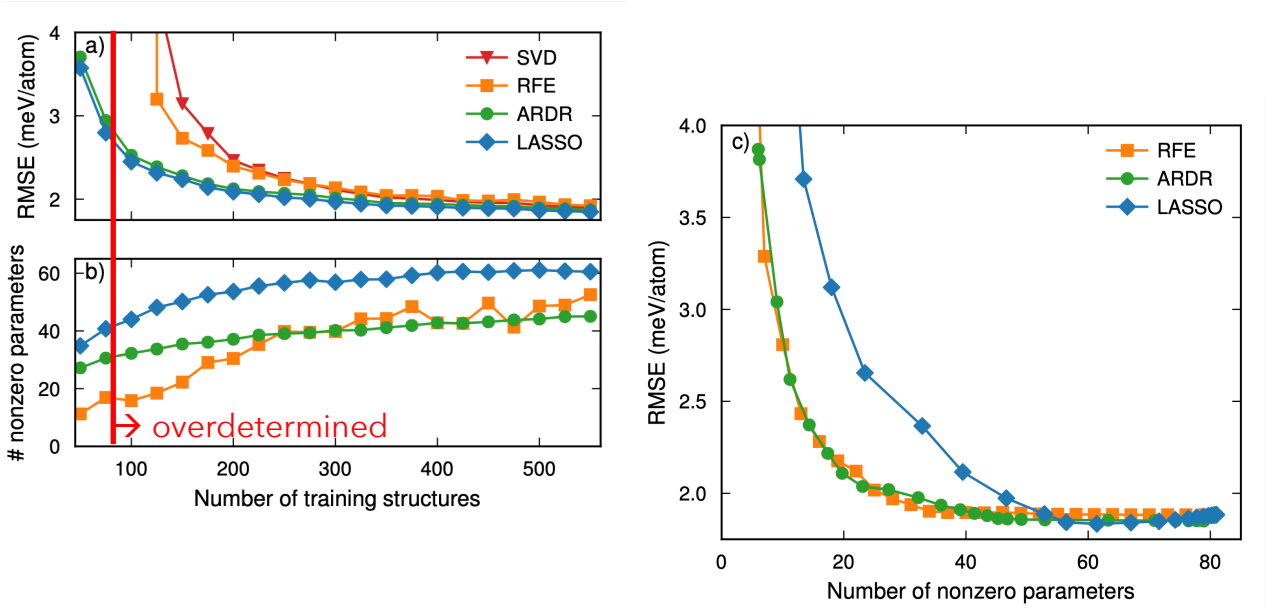
Solution: linear programming or **Huber regressor**

$$\mathcal{L} = H_\epsilon(\mathcal{D} - M(\boldsymbol{\theta})) + \dots$$

$$H_\epsilon(z) = \begin{cases} z^2 & \text{for} \quad |z| \leq \epsilon \\ 2\epsilon|z| - \epsilon^2 & \text{otherwise} \end{cases}$$

Method performance comparison

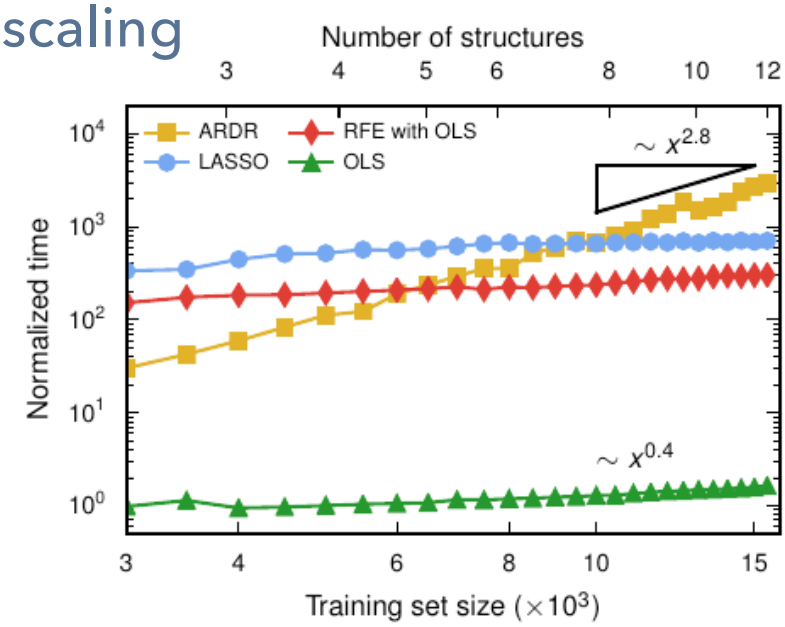
Example: Model for Ag-Pd alloy



Cluster expansion
→ Wednesday lecture

ARDR: fast convergence and sparse solution
LASSO: more false-positives
RFE: requires more structures

Computational scaling



Cost becomes significant for large problems
ARDR (Bayesian) becomes too costly due to scaling
RFE or OLS with manual selection often the most performant

Force constant expansion
→ lecture notes

Fransson *et al.*, npj Comp. Mater. **6**, 135 (2020)

Error correlation

Simulating reality

Data $\boldsymbol{\mathcal{D}} = \{d_i\}_{i=1}^{N_d}$ — Observations

$d_i = \zeta(\boldsymbol{x}_i) + \varepsilon_i$ — Measurement error

Control variables (temperature, angle, position…)

True or real value (unknown)

Model $\zeta(\boldsymbol{x}_i) = M(\boldsymbol{\theta}; \boldsymbol{x}_i) + \delta(\boldsymbol{x}_i)$

Model discrepancy

Parameter vector

Link between model and reality $d_i = M(\boldsymbol{\theta}; \boldsymbol{x}_i) + \varepsilon_i + \delta(\boldsymbol{x}_i)$

What did we do before?

Simulated data

$$\widehat{\boldsymbol{\mathcal{D}}} = \widehat{d}_i = M(\boldsymbol{\theta}; \boldsymbol{x}_i) + \delta(\boldsymbol{x}_i)$$

$$\chi^2 = \boldsymbol{\varepsilon}^T \boldsymbol{\varepsilon} \quad \text{with} \quad \boldsymbol{\varepsilon} = \widehat{\boldsymbol{\mathcal{D}}} - \boldsymbol{\mathcal{D}}$$

Loss function

Residual

Given data

→ Least-squares minimization

$$\boldsymbol{\theta}_{\text{opt}} = \arg\min_{\boldsymbol{\theta}} \chi^2$$

1. We did not *consciously* model the error
2. We did not use any prior knowledge regarding the error

Modeling more complex error patterns

Generalize the variance = go beyond i.i.d. errors

$$\frac{\boldsymbol{\varepsilon}^T \boldsymbol{\varepsilon}}{\sigma^2} \rightarrow \boldsymbol{\varepsilon}^T \boldsymbol{\Sigma}^{-1} \boldsymbol{\varepsilon}$$

Covariance matrix

$$p(\boldsymbol{\mathcal{D}}|\boldsymbol{\theta}, \sigma^2) = \left(\frac{1}{2\pi\sigma^2}\right)^{N_d/2} \exp\left\{-\frac{1}{2}\boldsymbol{\varepsilon}^T \boldsymbol{\Sigma}^{-1} \boldsymbol{\varepsilon}\right\}$$

$\boldsymbol{\Sigma} = \sigma^2 \mathbf{1}$

i.i.d.

$\boldsymbol{\Sigma} = \text{diag}$

e.g., error varies with 2θ or signal intensity

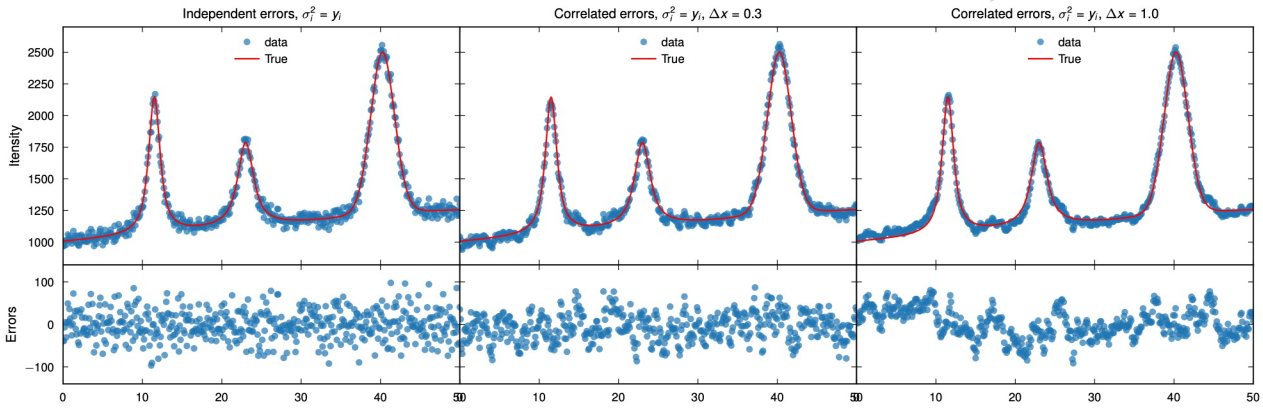
$\boldsymbol{\Sigma} = \text{full}$

e.g., correlated errors

Modeling more complex error patterns

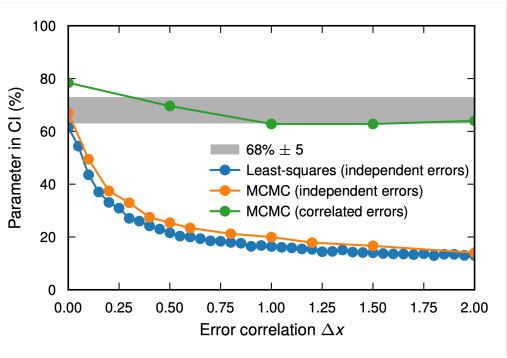
A demonstration

Increasing correlation



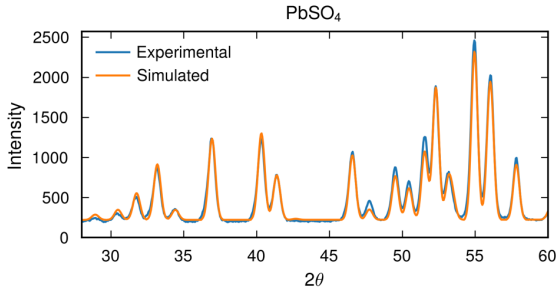
$$\Sigma_{ij} = \sigma^2 \exp(-|x_i - x_j|/\Delta x)$$

If error correlations are untreated the parameter uncertainty is dramatically underestimated



CHALMERS
UNIVERSITY OF TECHNOLOGY

Example: Refining diffraction data



$\boldsymbol{y}_{\text{exp}}$ Measured intensity

$\boldsymbol{y}_{\text{sim}}$ Simulated intensity

$\boldsymbol{x}$ Instrumental parameters, structural parameters etc.

Minimization problem

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \chi^2(\boldsymbol{x})$$

$$\chi^2(\boldsymbol{x}) = \sum_i \frac{1}{\sigma^2} \frac{(y_{i,\text{sim}} - y_{i,\text{exp}})^2}{y_{i,\text{exp}}^2}$$

Likelihood from assumption of identical normal errors

$$y_{i,\text{exp}} = y_{i,\text{sim}} + \varepsilon_i$$

$$\varepsilon_i \sim \mathcal{N}(0, \sigma^2 y_{i,\text{exp}}^2)$$

CHALMERS
UNIVERSITY OF TECHNOLOGY

MCMC vs Least-squares

$$p(\boldsymbol{x}) = \exp(-\chi^2(\boldsymbol{x}))$$

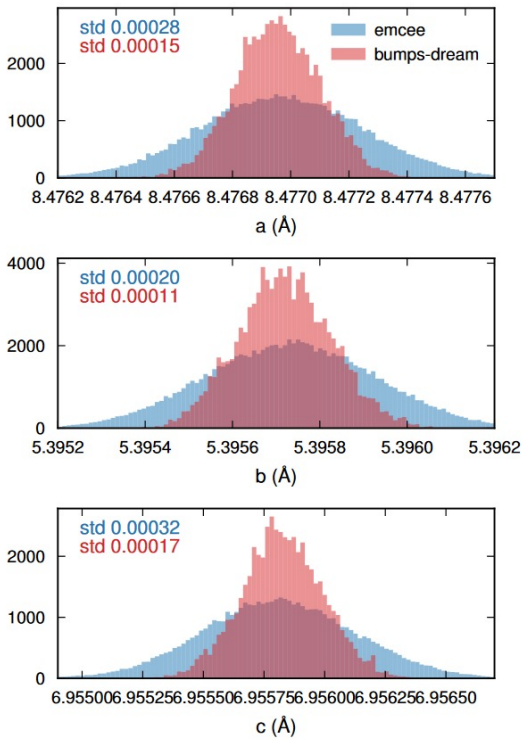
$$\chi^2(\boldsymbol{x}) = \sum_i \frac{1}{\sigma^2} \frac{\left(y_{i,\mathrm{sim}} - y_{i,\mathrm{exp}}\right)^2}{y_{i,\mathrm{exp}}^2}$$

least squares (lmfit)
a = 8.47701 ± 0.00027 Å
b = 5.39576 ± 0.00019 Å
c = 6.95584 ± 0.00032 Å

MCMC (emcee) σ≈2.0
a = 8.47697 ± 0.00028 Å
b = 5.39564 ± 0.00019 Å
c = 6.95585 ± 0.00031 Å

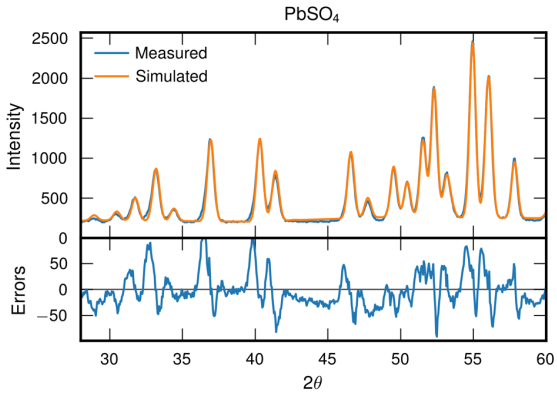
MCMC (bumps, dream; no correlation)
a = 8.47699 ± 0.00015 Å
b = 5.39571 ± 0.00011 Å
c = 6.95584 ± 0.00017 Å

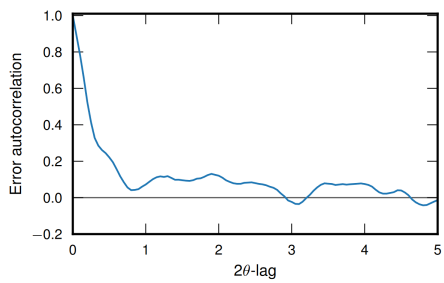
Smaller σ → Smaller errors
More data → Smaller errors



Error correlations

ε_i correlates with ε_j





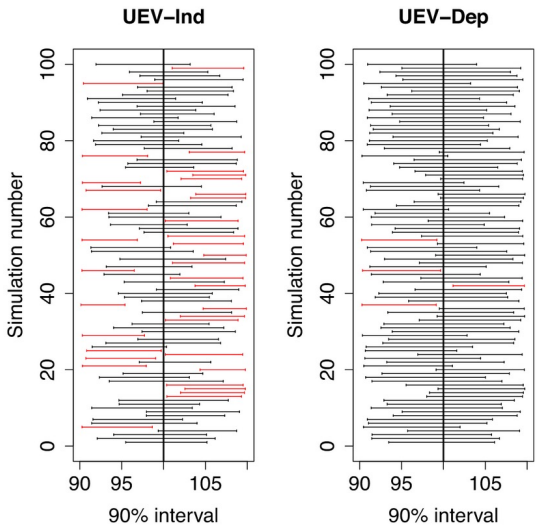
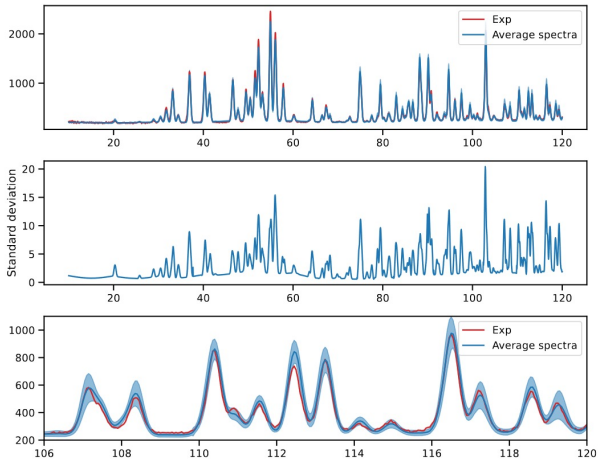


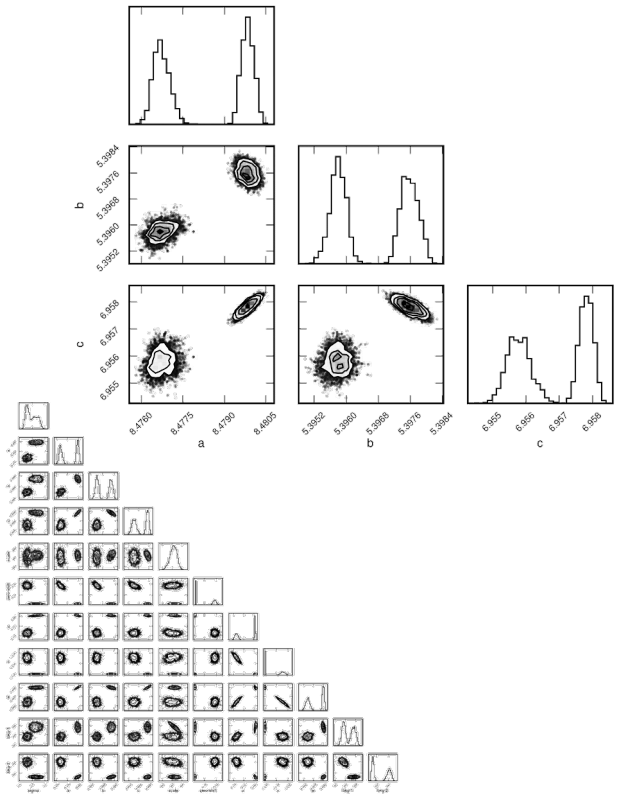
Figure 4. Coverage of the 90% interval. Illustration of the coverage in the simulation study. The plot shows the 90% posterior interval for $I_\circ$ for all 100 simulated data sets and the unequal variance models with independent (left) and dependent (right) residuals. The intervals that exclude the true value (vertical line at $I_\circ = 100$) are show in red. The coverage percentage reported in Table 2 is the percent of the 100 intervals that include the true value.

Fancher, C., Han, Z., Levin, I. *et al.*
Use of Bayesian Inference in Crystallographic
Structure Refinement via Full Diffraction Profile Analysis
Sci Rep **6,** 31625 (2016); https://doi.org/10.1038/srep31625

Ensemble of spectra

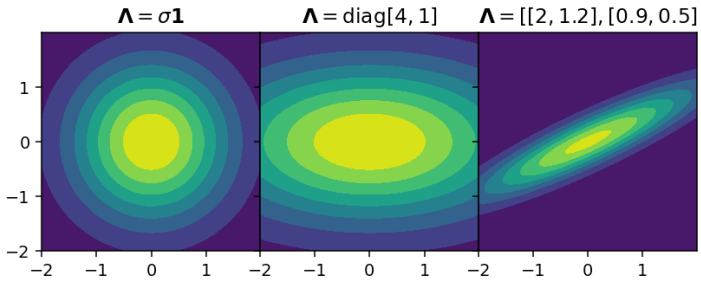


Multiple minima for independent MCMC walkers



CHALMERS
UNIVERSITY OF TECHNOLOGY

Take aways



$$\mathcal{L} = \hat{\varepsilon}^T \mathbf{\Sigma}^{-1} \hat{\varepsilon} + \boldsymbol{\theta}^T \mathbf{\Lambda}^{-1} \boldsymbol{\theta}$$

Model information

Error information

Schematic of relation between methods

Method	Likelihood	Parameter prior	Hyperparameter prior
OLS			
Ridge			
Bayesian ridge			
ARD			
LASSO			
Robust regression			

Error correlation