Challenges in building widely applicable GP software

Aki Vehtari

Aalto University

Finnish Center for Artificial Intelligence

Stan

ArviZ

My background

My background

- The first Gaussian process software packages I used
  - FBM by Radford Neal
  - Netlab by Christopher Bishop and Ian Nabney

My background

- The first Gaussian process software packages I used
  - FBM by Radford Neal
  - Netlab by Christopher Bishop and Ian Nabney
- I lead the development of GPstuff software (Matlab/Octave)
  - developed methods and published on GPs and approximate inference, model selection and applications

My background

- The first Gaussian process software packages I used
  - FBM by Radford Neal
  - Netlab by Christopher Bishop and Ian Nabney
- I lead the development of GPstuff software (Matlab/Octave)
  - developed methods and published on GPs and approximate inference, model selection and applications
- I'm part of Stan development team
  - probabilistic programming framework (language, autodiff, inference engine, interfaces, ecosystem)

My background

- The first Gaussian process software packages I used
  - FBM by Radford Neal
  - Netlab by Christopher Bishop and Ian Nabney
- I lead the development of GPstuff software (Matlab/Octave)
  - developed methods and published on GPs and approximate inference, model selection and applications
- I'm part of Stan development team
  - probabilistic programming framework (language, autodiff, inference engine, interfaces, ecosystem)
  - GPs support is not a priority, but will get eventually better

Some applications

- predicting concrete quality
- spatial epidemiology
- cancer recurrence risk prediction
- finding minimum energy paths and saddle points
- ABC / likelihood free inference
- medicine dosage design

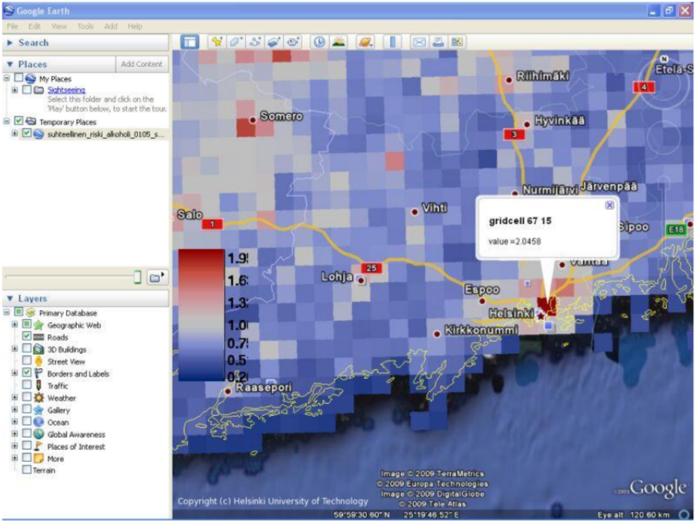
Some applications

- predicting concrete quality

- spatial epidemiology
- cancer recurrence risk prediction
- finding minimum energy paths and saddle points
- ABC / likelihood free inference
- medicine dosage design

Some applications

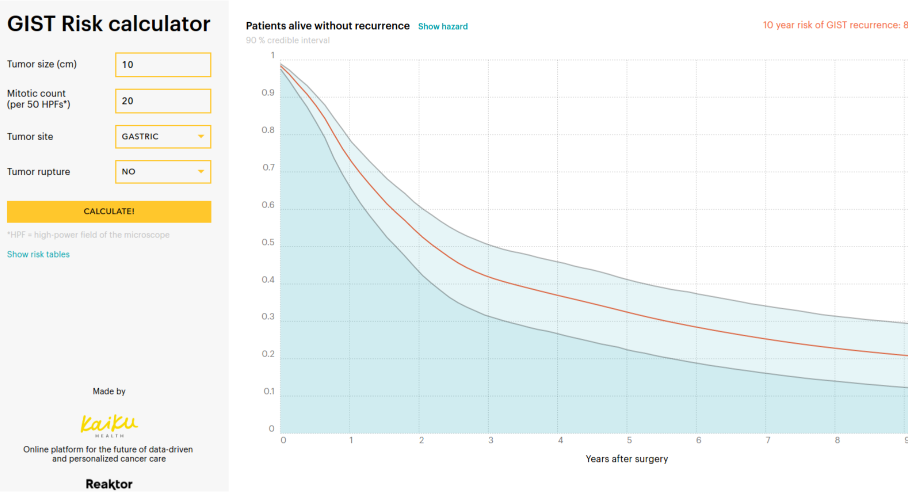
- predicting concrete quality
- spatial epidemiology



- cancer recurrence risk prediction

Some applications

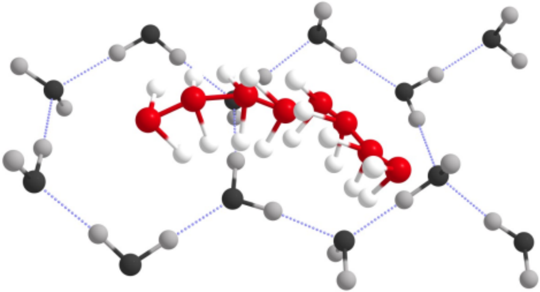
- predicting concrete quality
- spatial epidemiology
- cancer recurrence risk prediction



- finding minimum energy paths and saddle points

Some applications

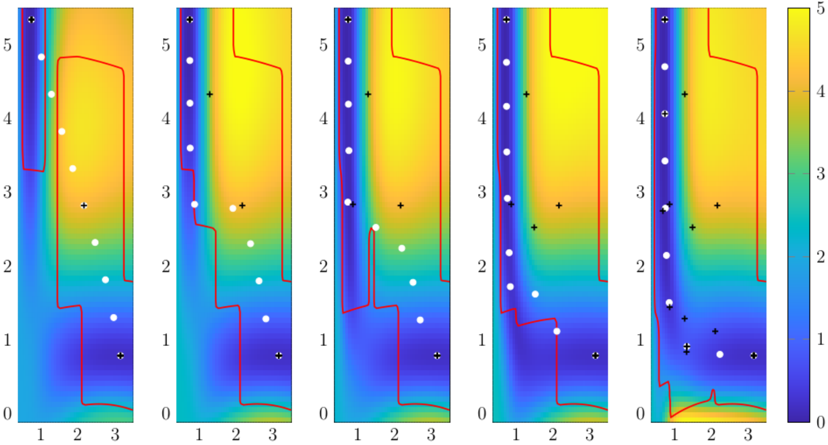
- predicting concrete quality
- spatial epidemiology
- cancer recurrence risk prediction
- finding minimum energy paths and saddle points



- ABC / likelihood free inference
- medicine dosage design

Some applications

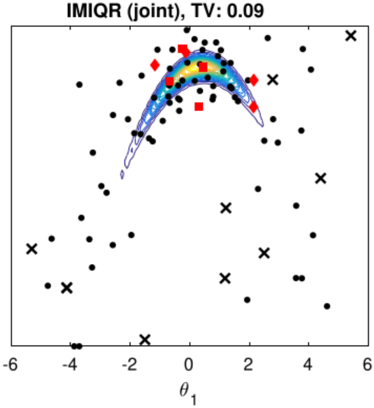
- predicting concrete quality
- spatial epidemiology
- cancer recurrence risk prediction
- finding minimum energy paths and saddle points



- ABC / likelihood free inference

Some applications

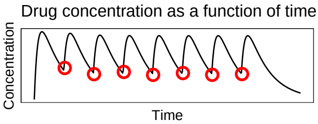
- predicting concrete quality
- spatial epidemiology
- cancer recurrence risk prediction
- finding minimum energy paths and saddle points
- ABC / likelihood free inference



- medicine dosage design

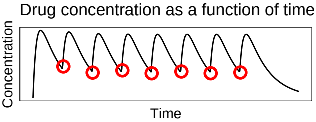
Some applications

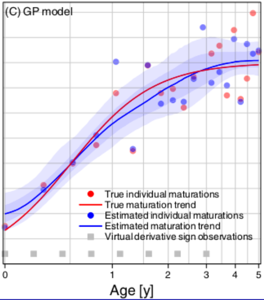
- predicting concrete quality
- spatial epidemiology
- cancer recurrence risk prediction
- finding minimum energy paths and saddle points
- ABC / likelihood free inference
- medicine dosage design



Some applications

- predicting concrete quality
- spatial epidemiology
- cancer recurrence risk prediction
- finding minimum energy paths and saddle points
- ABC / likelihood free inference
- medicine dosage design



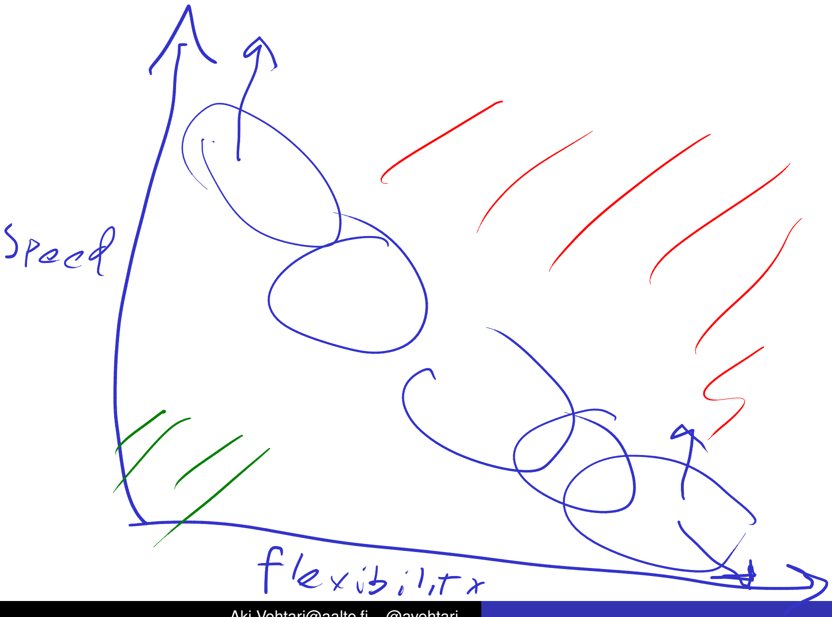


Some other software

- GPML
- GPy
- GPFlow
- GPtorch
- PyMC3
- TensorFlow probabilities
- Pyro
- Turing.JL
- INLA
- mgcv

https://en.wikipedia.org/wiki/Comparison_of_Gaussian_process_software

Flexibility vs speed



Speed

- Direct implementation for Gaussian-GP $O(n^3)$

Speed

- Direct implementation for Gaussian-GP $O(n^3)$
- Faster with specific algorithms, e.g.

Speed

- Direct implementation for Gaussian-GP $O(n^3)$
- Faster with specific algorithms, e.g.
  - 1D equispaced grid $O(n \log n)$
  - 1D Kalman filtering/smoothing $O(n)$
  - 1–4D basis functions

Speed

- Direct implementation for Gaussian-GP $O(n^3)$
- Faster with specific algorithms, e.g.
  - 1D equispaced grid $O(n \log n)$
  - 1D Kalman filtering/smoothing $O(n)$
  - 1–4D basis functions
  - Kroenecker

Speed

- Direct implementation for Gaussian-GP $O(n^3)$
- Faster with specific algorithms, e.g.
  - 1D equispaced grid $O(n \log n)$
  - 1D Kalman filtering/smoothing $O(n)$
  - 1–4D basis functions
  - Kroenecker
  - compact covariance function and sparse matrices
  - sparse precision matrices
  - low dimensional local/hierarchical covariance approximations

Speed

- Direct implementation for Gaussian-GP $O(n^3)$
- Faster with specific algorithms, e.g.
  - 1D equispaced grid $O(n \log n)$
  - 1D Kalman filtering/smoothing $O(n)$
  - 1–4D basis functions
  - Kroenecker
  - compact covariance function and sparse matrices
  - sparse precision matrices
  - low dimensional local/hierarchical covariance approximations
  - inducing point approaches

Speed

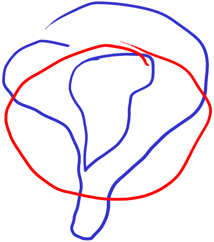
- Direct implementation for Gaussian-GP $O(n^3)$
- Faster with specific algorithms, e.g.
  - 1D equispaced grid $O(n \log n)$
  - 1D Kalman filtering/smoothing $O(n)$
  - 1–4D basis functions
  - Kroenecker
  - compact covariance function and sparse matrices
  - sparse precision matrices
  - low dimensional local/hierarchical covariance approximations
  - inducing point approaches
- Which approximation to use depends on e.g. stationarity, relative correlation length, combination of covariance functions

Speed

- The full joint posterior has difficult geometry
  - MCMC is likely to be slow
  - distributional approximations are likely to be bad
- The conditional distribution for latent values is easier
  - integrate out the latent variables using approximations
  - Laplace, EP, variational
  - if big data, maximizing marginal likelihood is ok
- Things get more difficult on the next slide

$$\underset{\sim}{f} \sim GP(0, K(x, x'; \hat{\underset{\sim}{\theta}}))$$

$$y \sim p(f, \hat{\underset{\sim}{\phi}})$$



Flexibility

- Different observation models
  - exponential family easy

Flexibility

- Different observation models
  - exponential family easy
  - non-exponential family varyingly difficult

Flexibility

- Different observation models
  - exponential family easy
  - non-exponential family varyingly difficult
  - observation models depending on multiple latent values
  - observation models depending on multiple observations
  - censored data
  - multioutput
  - derivative observations

Flexibility vs complexity

- Combinatorial explosion if all features need to work together
  - approximate computation related to covariance matrix
  - approximate integration (latent or joint)
  - different observation models
  - different priors
  - combine with other models like ODEs

Flexibility vs speed

- How about Turing complete probabilistic programming language, autodiff and automatic inference?

Flexibility vs speed

- How about Turing complete probabilistic programming language, autodiff and automatic inference?
- Speed in autodiff systems is not automatic!

Flexibility vs speed

- How about Turing complete probabilistic programming language, autodiff and automatic inference?
- Speed in autodiff systems is not automatic!
  - what is a node in autodiff?
  - forward, reverse, mixed, adjoints, etc.

Flexibility vs speed

- How about Turing complete probabilistic programming language, autodiff and automatic inference?
- Speed in autodiff systems is not automatic!
  - what is a node in autodiff?
  - forward, reverse, mixed, adjoints, etc.
- Inference speed depends on
  - computational cost of single (marginal) log density
  - difficult posterior geometries require more (marginal) log density evluations
  - integration vs maximzing marginal likelihood

GPs and Stan

- For 1D-3D we recommend basis functions

GPs and Stan

- For 1D-3D we recommend basis functions
- Kalman filtering/smoothing exists, but slow without explicit derivatives

GPs and Stan

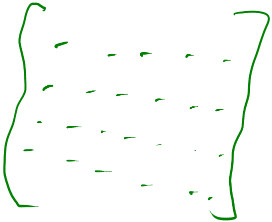
- For 1D-3D we recommend basis functions
- Kalman filtering/smoothing exists, but slow without explicit derivatives
- For small data covariance matrix approach with $O(n^3)$ feasible

GPs and Stan

- For 1D-3D we recommend basis functions
- Kalman filtering/smoothing exists, but slow without explicit derivatives
- For small data covariance matrix approach with $O(n^3)$ feasible
- Matrix variable coming soon

n^2

n^3



K[i,j] = a;

GPs and Stan

- For 1D-3D we recommend basis functions
- Kalman filtering/smoothing exists, but slow without explicit derivatives
- For small data covariance matrix approach with $O(n^3)$ feasible
- Matrix variable coming soon
- Sparse matrices coming

GPs and Stan

- For 1D-3D we recommend basis functions
- Kalman filtering/smoothing exists, but slow without explicit derivatives
- For small data covariance matrix approach with $O(n^3)$ feasible
- Matrix variable coming soon
- Sparse matrices coming
- Laplace integration over the latents work in progress

GPs and Stan

- For 1D-3D we recommend basis functions
- Kalman filtering/smoothing exists, but slow without explicit derivatives
- For small data covariance matrix approach with $O(n^3)$ feasible
- Matrix variable coming soon
- Sparse matrices coming
- Laplace integration over the latents work in progress
- Even with these, Stan (or other generic PPL frameworks) is not competing with specialized software

Conclusion

- Very unlikely that one software would be best for everything
- Tradeoff between flexibility, speed, and additional implementation effort
- Prediction: There will be improvements in modularity and interoperability