# QUESO

Damon McDougall

Centre for Predictive Engineering and Computational Sciences
Institute for Computational Engineering and Sciences
The University of Texas at Austin

6th of April, 2016



# Outline

# QUESO

Nutshell: QUESO gives samples from $\mathbb{P}(\theta|y)$ (called MCMC)

- Library for Quantifying Uncertainty in Estimation, Simulation and Optimisation
- Born in 2008 as part of PECOS PSAAP programme
- Provides robust and scalable sampling algorithms for UQ in computational models
- Open source
- C++
- MPI for communication
- Parallel chains, each chain can house several processes
- Dependencies are MPI, Boost and GSL. Other optional features exist
- http://libqueso.com

# Contributors

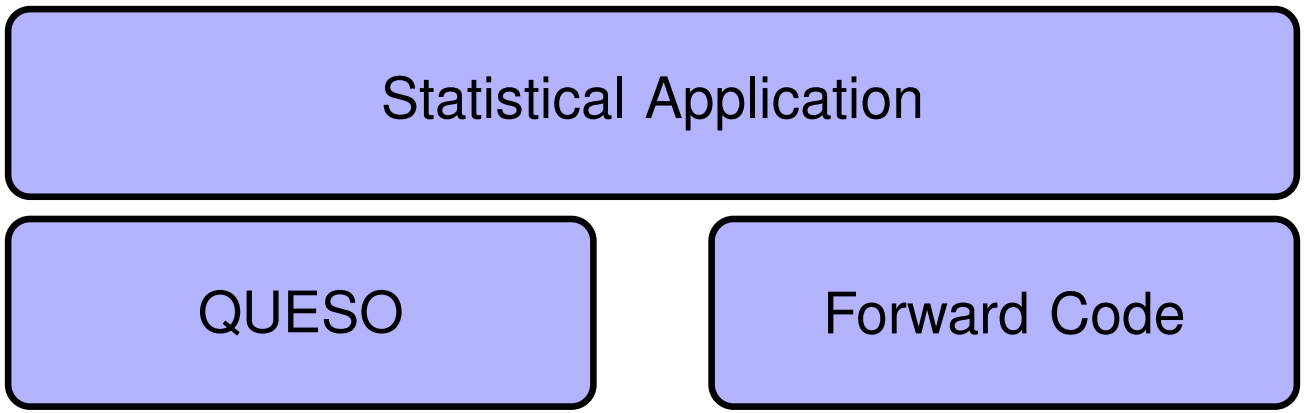
Brian Adams
Paul Bauman
Morgan Bruns
Sai Hung Cheung
Kemelli Estacio-Hiroms
Nicholas Malaya
André Maurente
Damon McDougall
Kenji Miki
Rebecca Morrison
Todd Oliver
Sylvain Plessis
Teresa Portone
Ernesto Prudencio
Karl Schulz
Roy Stogner
Gabriel Terejanu
Rhys Ulerich
Rochan Upadhyay
Eric Wright

# Why use QUESO?

Other solutions are available, e.g. R, PyMC, emcee, MICA, Stan, BET, SG++, UQTk, Opencossan, MUQ, Chaospy, Dakota, SmartUQ, Uqlab, PSUADE, UQ-PyL.

QUESO solves the same problem, but:

- Has been designed to be used with large forward problems
- Has been used successfully with 5000+ cores
- Leverages parallel MCMC algorithms
- Supports for finite **and** infinite dimensional problems



# Preliminaries

- Instructions on how to build and install QUESO: http://libqueso.com
- All QUESO docs are here: http://libqueso.com/queso/html/.
- Links on these slides are clickable

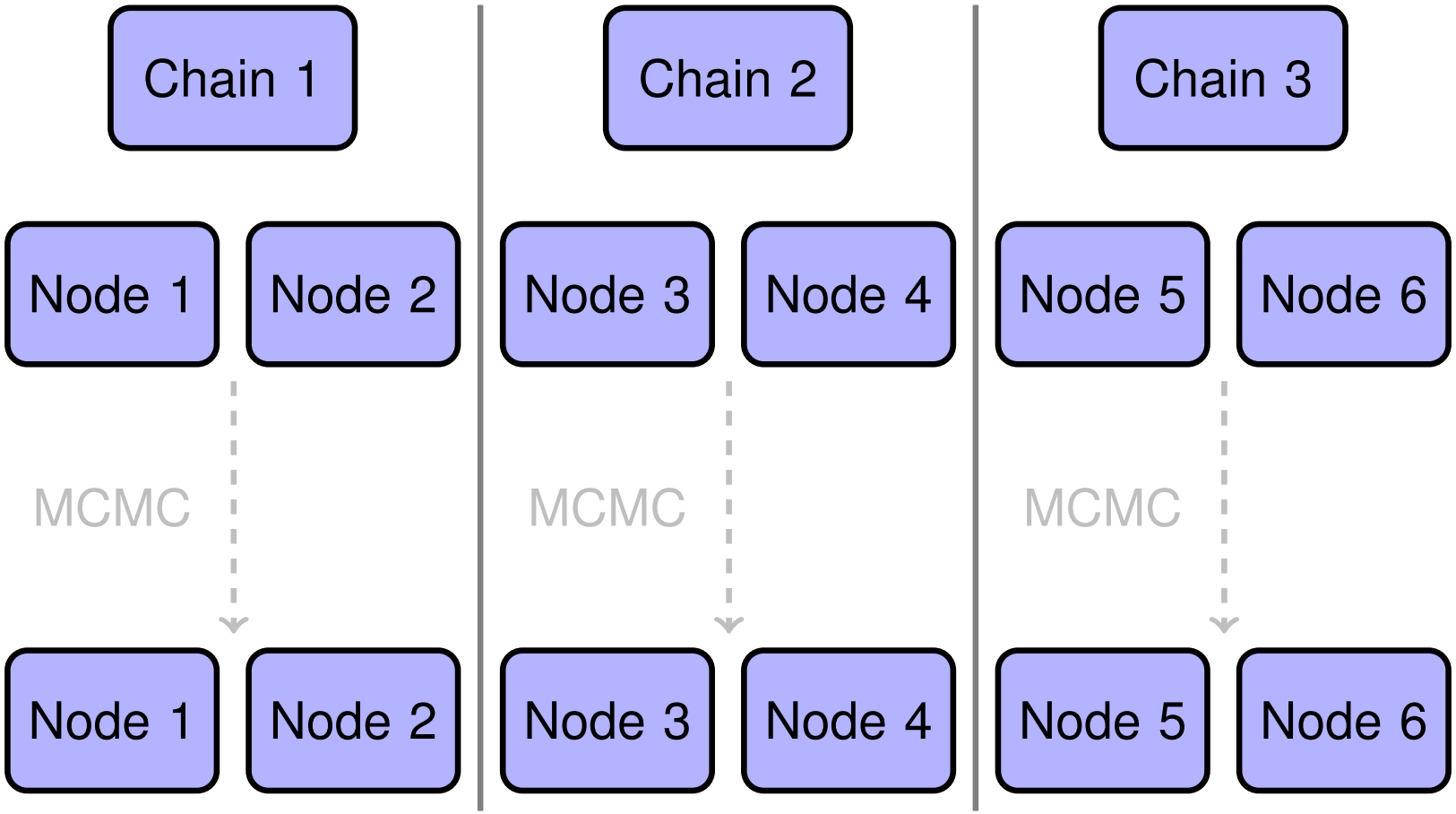
# What are the current properties of available tools?

- DRAM (Haario et al)
- Supports infinite dimensional problems (with `LibMesh`, different API)
  - pCN (Cotter et al)
- Leverages parallel MCMC algorithms (embarassingly parallel)
- Chain output formats: Matlab, tabular ASCII, HDF5
- (Very simple) Surrogates:
  - Linear interpolation
  - GPMSA (Brian Williams, Brian Adams, Ralph Smith)
- Aim to drive flexibility from an input file (no re-compile)
- Rely on Boost for C++11 foo
- Leverages deterministic optimisation for initial chain position

# For which classes of problems have they been developed?

- Big ones

# Parallel QUESO



# Examples

- What does a statistical application look like?
  - `https://github.com/libqueso/queso/blob/dev/examples/template_example/template_example.cpp`
- What does the input file look like?
  - `https://github.com/libqueso/queso/blob/dev/examples/template_example/template_example_input.txt`
- What does the output look like?
  - `https://github.com/libqueso/queso/blob/dev/test/test_Regression/test_gpmsa_cobra_samples.m`

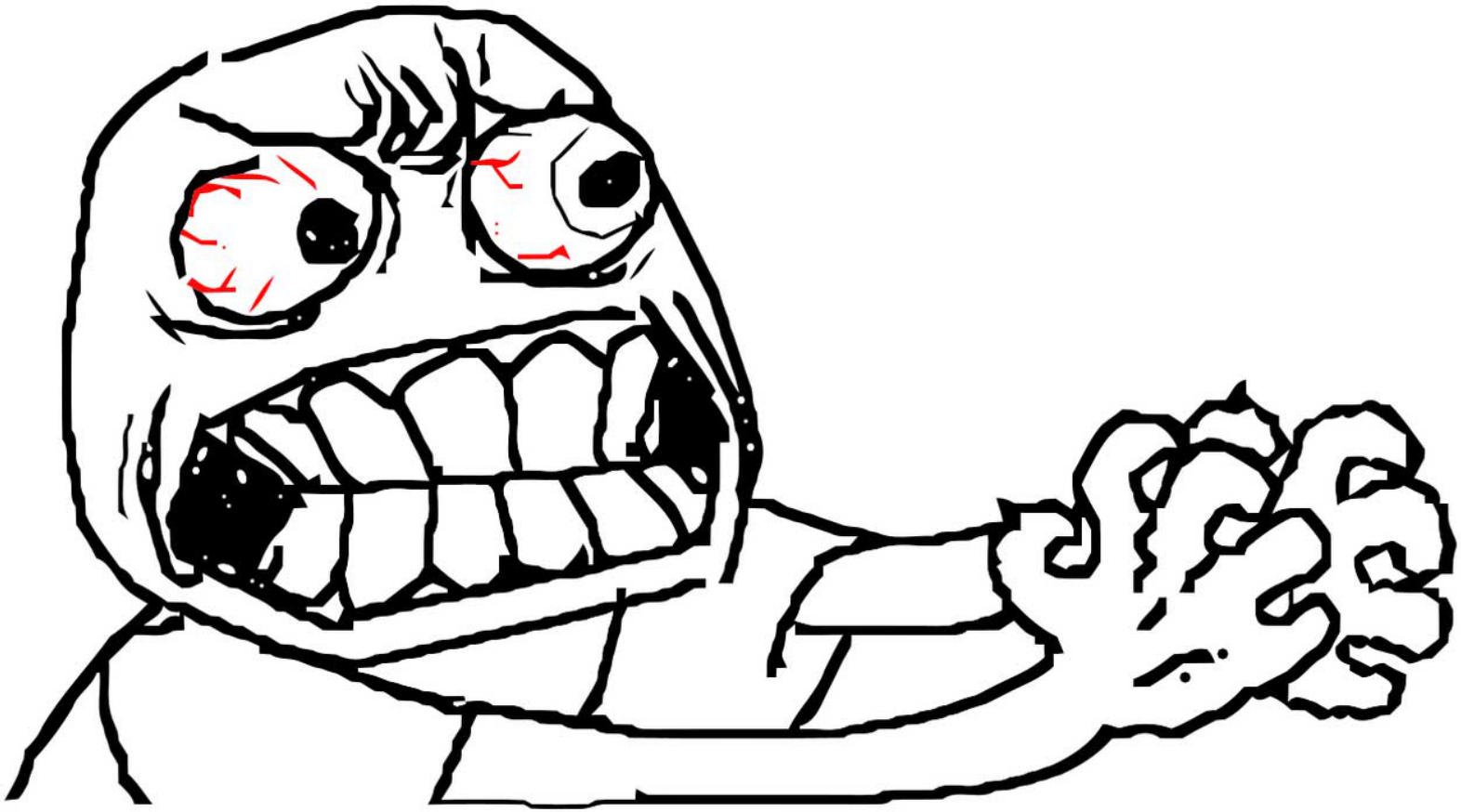
# What are challenges for UQ software and which resources are required?

- Getting funding for software is hard
  - Scientific Software Days: `http://scisoftdays.org`
- Dimensionality
  - Active subspaces (Constantine et al)
  - Likelihood-informed subspace (Cui et al)
  - Where should pre-processing live? In UQ software? On top of UQ software?
- Optimise before sampling
- Software best practices: version control, semantic versioning, backwards compatibility
- Provide the gradient (and Hessian action) of your model at a point

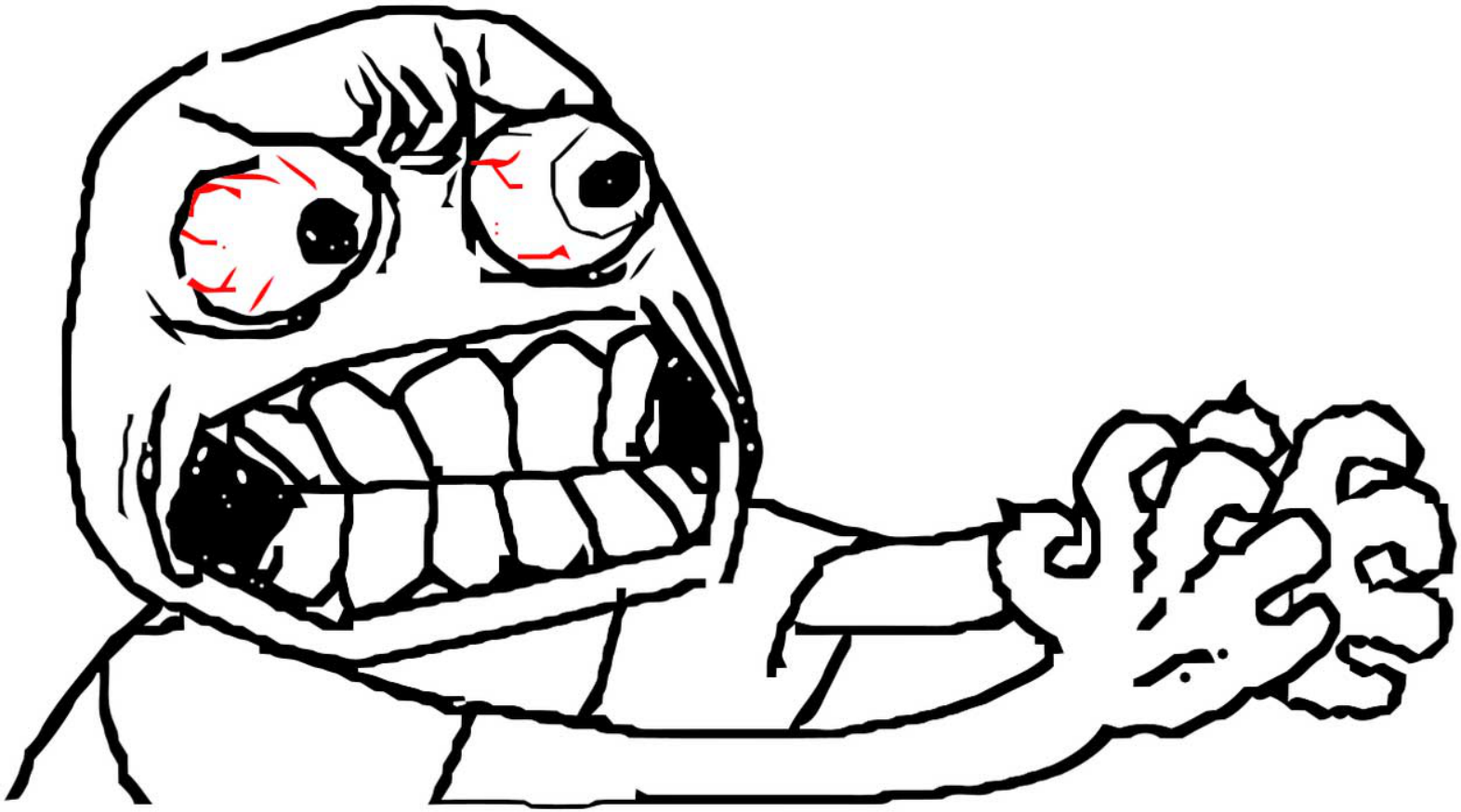
# What are challenges for UQ software and which resources are required?

- Verification: how do I test?
  - CLT?
  - Point values of PDFs? Need to know normalising constant.
  - Anderson-Darling?
  - Kolmogorov-Smirnov? Higher dimensions?
  - Benchmark problems a la deterministic optimisation?
    - The Rosenbrock function for Bayesian inference?
  - Moments
    - Integration is hard. Riemann? Sparse grid?

What do I do when my tests fail?



What do I do when my tests *never* fail?



# What are the next steps and the development goals?

- Actually parallel: chains can swap statistics
- Actually C++11
  - If the compiler isn't ancient, don't require Boost
- API consistency for infinite dimensional problems
  - And without a `LibMesh` requirement
- Surrogates?
- Focus?
  - Do sampling and do it well?
  - Provide a mechanism for enabling UQ algorithms research?
  - Both?

# What are the next steps and the development goals?

- QUESO works closely with the Dakota team at Sandia
  - They provide feedback on our design
  - They request features
  - They find bugs
  - We learn from their (software and optimisation) expertise
  - Dakota can link to QUESO for inference
  - Dakota has a large userbase
- We have funding from Sandia to:
  - Improve the sampler interface to improve and ease maintainability for new and existing MCMC algorithms
  - Implement a generic vector/matrix interface so we can have higher performance vector implementations
  - Usability, user-friendliness, and licensing improvements

# Contributing

- All software has bugs
- QUESO is open source software, so it is free
- Free means you don't have to pay for it (beer)
- Free also means you can play with the source code yourself (speech)
- If you use QUESO and you need help, please email the users mailing list
  - queso-users@googlegroups.com
- To stay apprised of new releases or discuss future directions email the developers list
  - queso-dev@googlegroups.com
- You can also help others that ask questions on these lists

# Contributing

- If you use QUESO and you encounter a bug, open a ticket
  - `https://github.com/libqueso/queso/issues/new`
- If you fix a bug and want to contribute to the QUESO codebase
  - Awesome
  - Review contribution guidelines here: `https://github.com/libqueso/queso#contributing`
  - Write tests for your patches
  - And if you add a new feature, make sure to write documentation

# Questions?