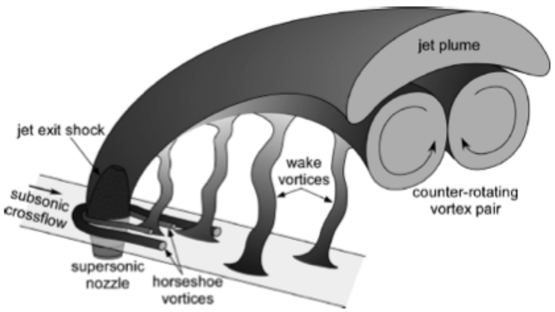
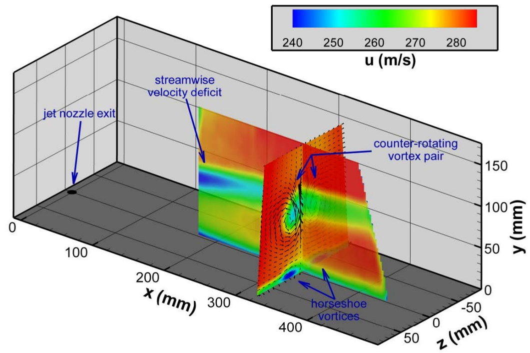




Using LASSO to infer a high-order eddy viscosity model for k-ε RANS simulation of transonic flows

S. Lefantzi, **J. Ray,** S. Arunajatesan and L. Dechant

(jairay@sandia.gov)

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000. SAND2014-2429C

SAND2016-4661C

The problem

- Aim: Develop a predictive k-ε RANS model for transonic jet-in-crossflow (JinC) simulations
- Drawback: RANS simulations are simply not predictive
  - They have "model-form" error i.e., missing physics
  - The numerical constants/parameters in the k-ε model are usually derived from canonical flows
- Hypothesis
  - One can calibrate RANS to jet-in-crossflow experiments; thereafter the residual error is mostly model-form error
  - Due to model-form error and limited experimental measurements, the parameter estimates will be approximate
    - We will estimate parameters as probability density functions (PDF)
  - We then address the model-form error with an enriched eddy viscosity model for the missing physics

The equations

- The model
  - Devising a method to calibrate k-ε parameters from expt. data

$$\frac{\partial \rho k}{\partial t}+\frac{\partial}{\partial x_i}\left[\rho u_i k-\left(\mu+\frac{\mu_T}{\sigma_k}\right)\frac{\partial k}{\partial x_i}\right]=P_k-\rho\varepsilon+S_k$$

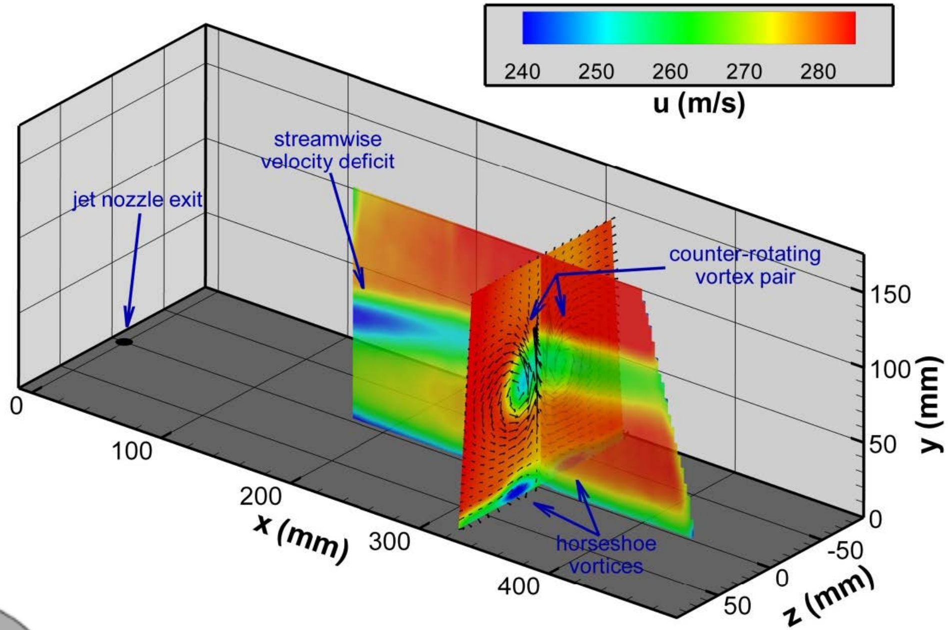
$$\frac{\partial \rho \varepsilon}{\partial t}+\frac{\partial}{\partial x_i}\left[\rho u_i \varepsilon-\left(\mu+\frac{\mu_T}{\sigma_\varepsilon}\right)\frac{\partial \varepsilon}{\partial x_i}\right]=\frac{\varepsilon}{k}\left(C_1 f_1 P_k-C_2 f_2\rho\varepsilon\right)+S_\varepsilon$$

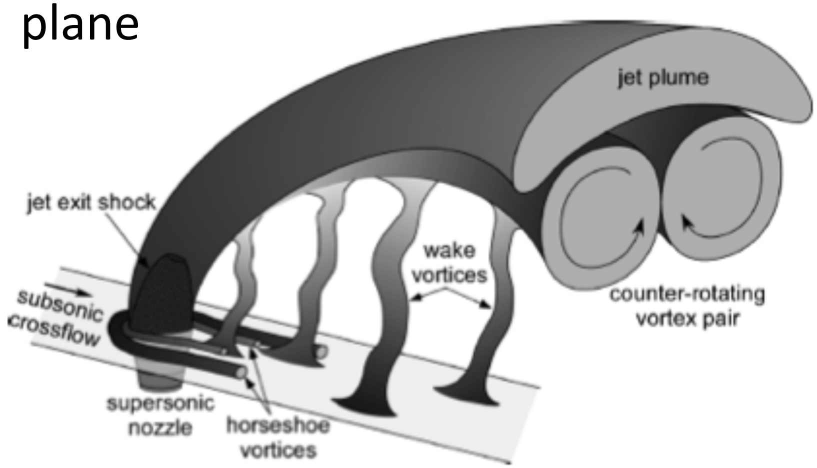
$$\mu_T=C_\mu f_\mu \rho\frac{k^2}{\varepsilon}$$

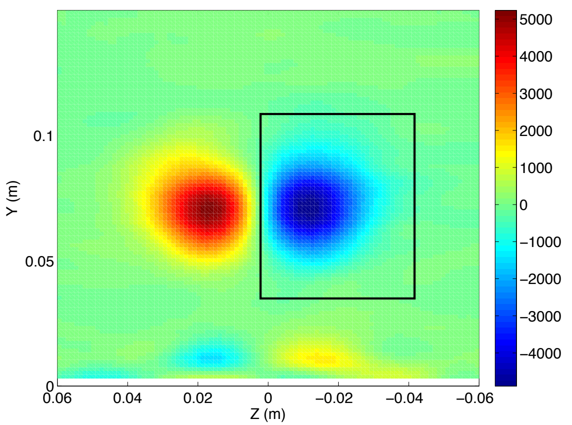
- Sources of errors
  - Parameters $\{C_2,\ C_1\}$ are obtained from canonical flows
  - C_μ is deemed constant throughout the flowfield
  - Linear stress-strain rate relationship $\tau_{ij} = -2/3k\ \delta_{ij} + \mu_T\ S_{ij}$
    - Called a linear eddy viscosity model (LEVM)

Target problem - jet-in-crossflow

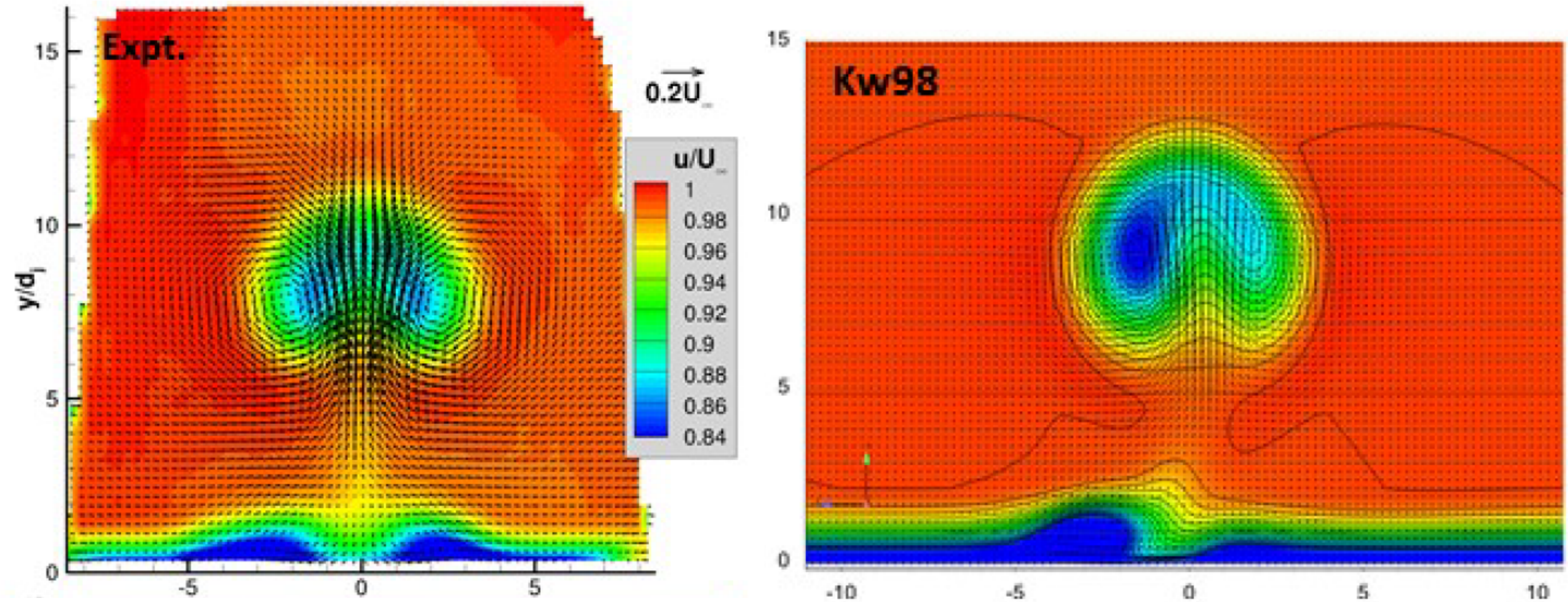
- A canonical problem for spin-rocket maneuvering, fuel-air mixing etc.
- We have experimental data (PIV measurements) on the cross- and mid-plane
- Will calibrate to vorticity on the crossplane and test against mid-plane







RANS (k-ω) simulations - crossplane results



- Crossplane results for stream
- Computational results (SST) are too round; Kw98 doesn't have the mushroom shape; non-symmetric!
- Less intense regions; boundary layer too weak

Reducing errors

- Model-form errors
  - The linear turbulent stress – strain rate relationship (LEVM) can be enriched with quadratic and cubic terms (QEVM / CEVM)
    - Includes terms with vorticity and cross terms with vorticity and strain rate
  - However the high-order models have parameters in them
    - What are the appropriate values for those parameters?
- Parametric uncertainty
  - (C_2, C_1) can be estimated (somewhat) from experimental data
    - But because of model-form errors and limited experimental data, these cannot be estimated with much certainty
  - We'll use Bayesian inversion and estimate them as PDFs
    - Quantifies uncertainty in the estimate of the parameters
- Calibration process
  - Identify which of the CEVM parameters can actually be estimated from experimental data
  - Then calibrate those along with (C_2, C_1); call the full set $\mathbf{C} = (:, C_2, C_1)$

Calibration details

- Aims of the calibration
  - Calibrate to a M = 0.8, J = 10.2 interaction
  - Learn the form of the high-order eddy viscosity model by fitting to turbulent stresses measurements on the mid-plane
  - Calibrate to crossplane data; check by matching the midplane velocity profiles
- Technical challenges
  - Computational cost of 3D JinC RANS simulation
    - Replace 3D RANS with a surrogate model i.e., model crossplane streamwise vorticity $\omega^{(RANS)}_x(\mathbf{y}) = f(\mathbf{y}; \mathbf{C})$, $f(:; \mathbf{C})$ is a curve-fit
    - Surrogate model = emulators
  - Arbitrary combinations of **C** may be nonphysical
    - How to build emulators when **C** are nonsensical?
  - What functional form to use for $f(:; \mathbf{C})$?

High-order eddy-viscosity model

- Craft 95 describes a cubic eddy viscosity (CEVM) model
  - $\tau_{ij} = -2/3k\ \delta_{ij} + C_\mu F(S_{ij}, \varepsilon) + c_1 f_1(S_{ij}, \Omega_{ij}, \varepsilon) + c_2 f_2(S_{ij}, \Omega_{ij}, \varepsilon)$ ….. $c_7 f_7(S_{ij}, \Omega_{ij}, \varepsilon)$
  - $F(S_{ij})$ is linear in S_{ij}, $f_1(:, :, :)$ - $f_3(:, :, :)$ are quadratic in S_{ij} & Ω_{ij}
  - $f_4(:, :, :)$ – $f_7(:, :, :)$ are cubic in S_{ij} & Ω_{ij}
- Our experimental data, on the midplane, consists of:
  - S_{ij} & Ω_{ij} obtained from the measured velocity field
  - τ_{ij} and k, also measured
  - ε (dissipation rate of turbulent KE) cannot be measured
    - It is approximated by assuming equilibrium of production and dissipation of turbulent KE.
- Craft's model prescribes $\{c_1 \ldots c_7\}$
  - Parameter value obtained from a simple, incompressible turning flow
  - May not be valid for transonic JinC interaction

Estimation of CEVM parameters

- The 180 measurements that we have may not have info that informs $c_1 \dots c_7$
- Cast the estimation problem as

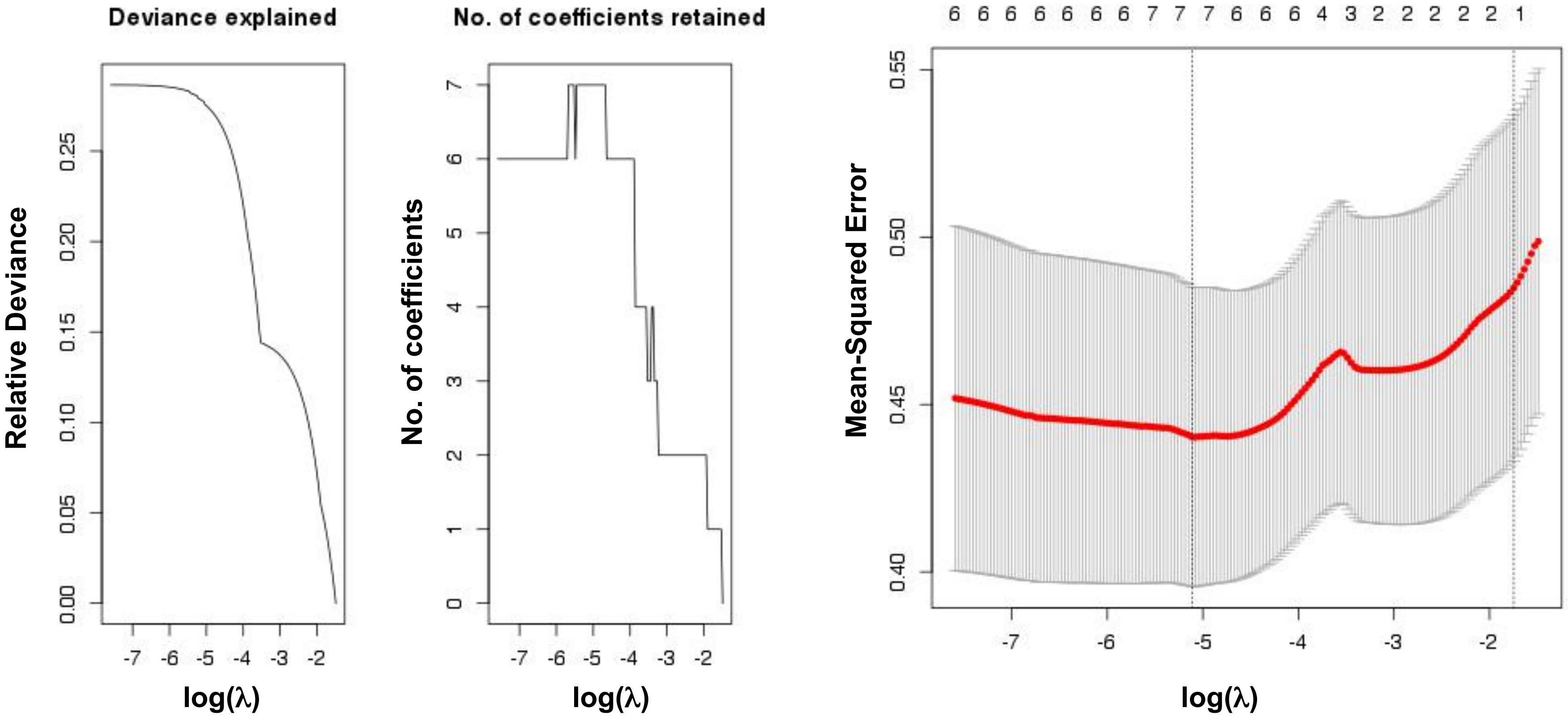
$$\boxed{\min_x \quad \|Y - Ac\|_2^2 + \lambda \|c\|_1}$$

  - The first half estimates x = $\{c_i\}$ that provide CEVM predictions near Y
  - The second half – the λ penalty – tries to set as many c_i to zero
  - Called Shrinkage Regression
- The penalty λ is the lynchpin
  - If it is too small, we get over-fitting (too many c_i survive)
  - The best way to get λ is via k-fold cross-validation
- The method for solving the optimization problem is LASSO

k-fold cross-validation

- Divide the 180 measurements into 8 “folds” (equal subsets)
- Pick a value of λ’
  - Pick fold # 1 as the testing set, folds 2-8 as the learning set
  - Solve the optimization problem (solve for **c**) using **Y** constructed from the learning set
  - Predict the data in the testing set
  - Repeat with folds #2, #3 … as the testing sets
  - Obtain the mean error and error bars for λ’
- Ultimately you get error as a function of λ
  - Pick the λ with min error
- For higher values of λ, expect to see lots of c_i becoming zero
  - And predictive errors becoming large
- Nomenclature: The norm of difference ($\mathbf{Y}^{(obs)} - \mathbf{Ac}$) is called the ‘deviance’

LASSO results



Looking for a good λ

- Craft explains around 28% of deviance
- As log(λ) increases and # of terms retained decreases, CEVM worsens
- One gets λ_{min} and λ_{1se}

Tabulate coefficients and MSE

Method	c_1	c_2	c_3	c_4	c_5	c_6	c_7	MSE
Craft	-0.1	0.1	0.26	-10	0	-5	5	0.662
λ_{min}	-0.065	-0.103	1.68	-4.02	5.7	5.4	-3.64	0.386
λ_{1se}	0.0	0.0	0.455	0.0	0	0	0	0.483
LM	-0.0789	-0.149	2.02	-5.88	0	6.68	-11.87	0.382

- $\ln(\lambda_{min}) = -5.11$, $\ln(\lambda_{1se}) = -1.75$
- Craft's default parameters are changed when we regress it to data
  - Results called 'LM'
- When we LASSO the model using λ_{1se}, we're left with just 1 quadratic term
  - But the model loses much accuracy
- Let's choose λ_{1se}.
  - Provides a simple model, and keeps the Ω^2 term

Calibration of $\{c_3, C_2, C_1\}$

- We will calibrate **C** = (c_3, C_2, C_1)
  - Our model really has a quadratic eddy viscosity model (QEVM)
- Approach:
  - Data: Use vorticity measurements on crossplane to estimate **C**
    - Useful measurements available at 225 locations ("probes")
  - Estimation procedure: Bayesian calibration using MCMC
  - Model: Use surrogate models (emulators) of the RANS simulator
    - Set of 1275 runs in the parameter space C to make the training data
    - Identify a physically realistic space $\mathcal{R}$, use SVMs to model $\mathcal{R}$
    - Make emulators $\omega(\mathbf{C}) = f(c_3, C_2, C_1)$ with polynomials; valid in $\mathcal{R}$
    - Use MCMC to create the posterior PDF of **C**
  - Checking results
    - Draw 100 samples from the posterior PDF
    - Develop an ensemble of predictions of vorticity and velocity; compare against measurements

The Bayesian calibration problem

- Model experimental values at probe j as $\omega^{(j)}_{ex} = \omega^{(j)}(\mathbf{C}) + \varepsilon^{(j)}$, $\varepsilon^{(j)} \sim N(0, \sigma^2)$

$$\Lambda\left(\omega_{ex}^{(j)} \mid C\right) \propto \prod_{j \in \mathcal{P}} \exp\left(- \frac{\left(\omega_{ex}^{(j)} - \omega^{(j)}(C)\right)^2}{2\sigma^2} \right)$$

- Given prior beliefs π on **C**, the posterior density ('the PDF') is

$$P(C, \sigma \mid \omega_{ex}^{(j)}) \propto \Lambda(\omega_{ex}^{(j)} \mid C, \sigma)\, \pi(c_3, C_2, C_1)\, \pi_\sigma(\sigma)$$

- $P(\mathbf{C}|\omega_{ex})$ is a complicated distribution that has to be described/ visualized by drawing samples from it
- This is done by MCMC
  - MCMC describes a random walk in the parameter space to identify good parameter combination
  - Each step of the walk requires a model run to check out the new parameter combination

Making emulators - 1

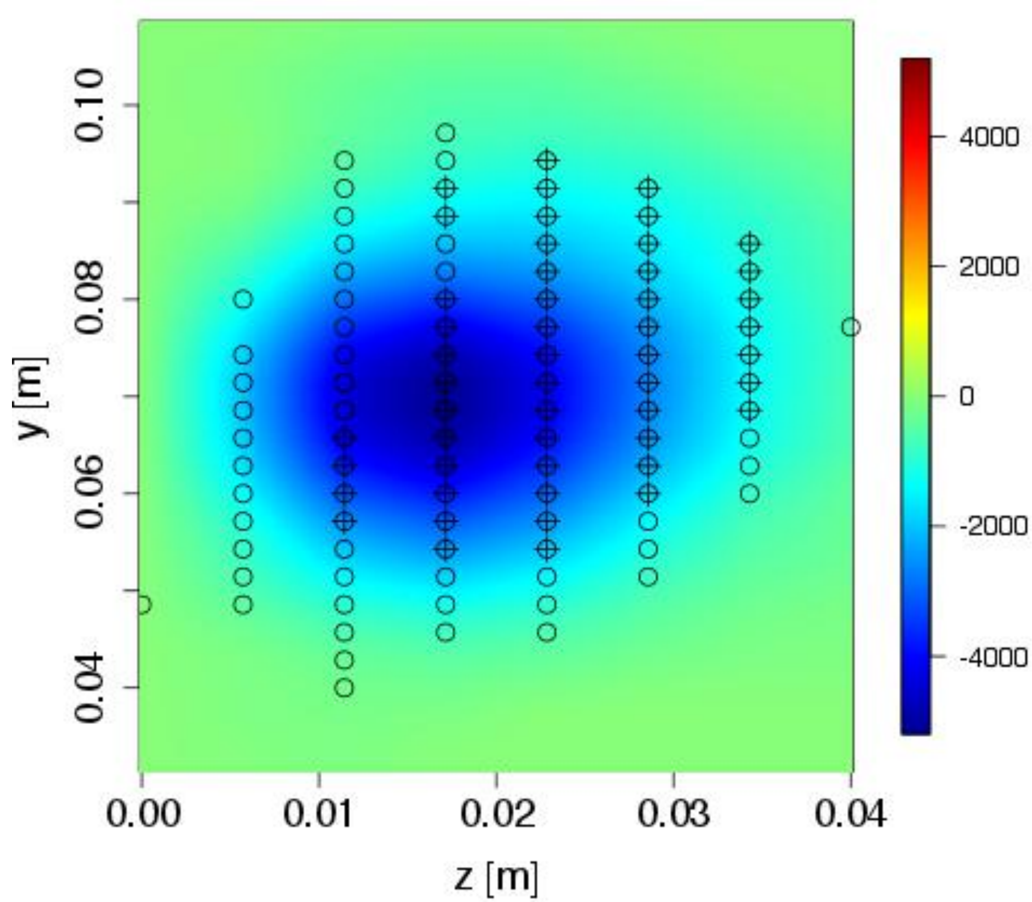
- Training data
  - Sample the parameter space **C** = {c_3, C_2, C_1}; bounds are known
  - Run RANS models at 1275 samples; save vorticity on cross-plane
  - Select the top 25% of the training runs
    - Call this subspace of $\mathcal{R}$
    - Keeps us out of non-physical parts of the parameter space C
- Making emulators in $\mathcal{R}$
  - Model vorticity at probe *j* $\omega^{(j)}$ as a polynomial in **C**

$$\omega^{(j)} \cong a_0 + a_1 c_3 + a_2 C_2 + a_3 C_1 + a_4 c_3 C_2 + a_5 c_3 C_1 + a_6 C_2 C_1 + \ldots..$$

  - Simplify using AIC; cross validated using repeated random sub-sampling (100 rounds)
    - RMSE in Learning & Testing sets should be equal
  - Accept all surrogate models that have < 10% error

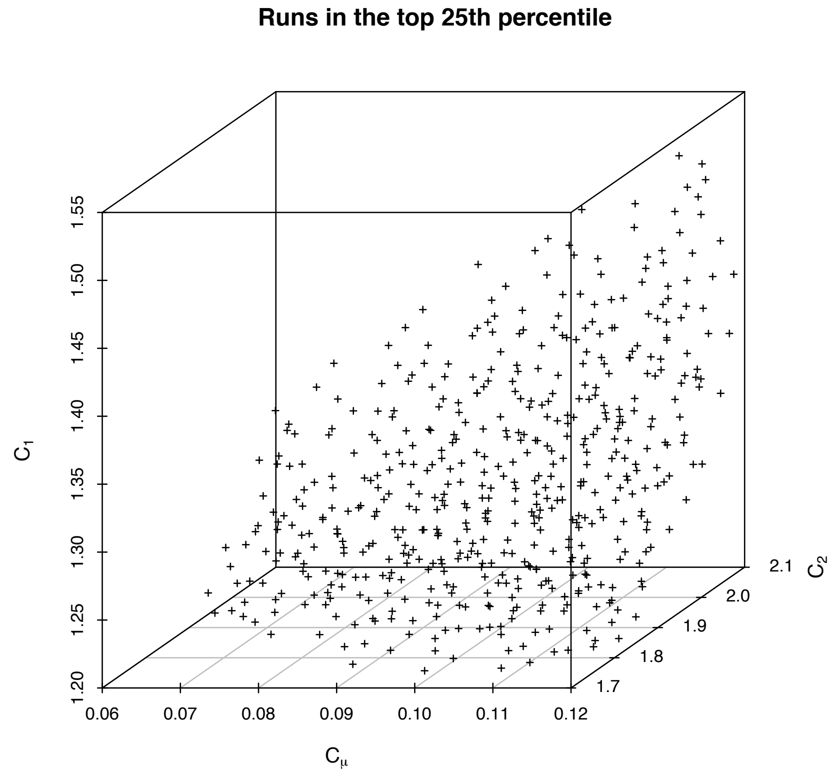
Making emulators - 2

- Emulators with 10% accuracy could only be made for 55 / 224 probes
  - 90 with large vorticity (circles)
  - 55 with emulators (+)
- Also, the emulators are only applicable in the $\mathcal{R}$ section of the parameter space C



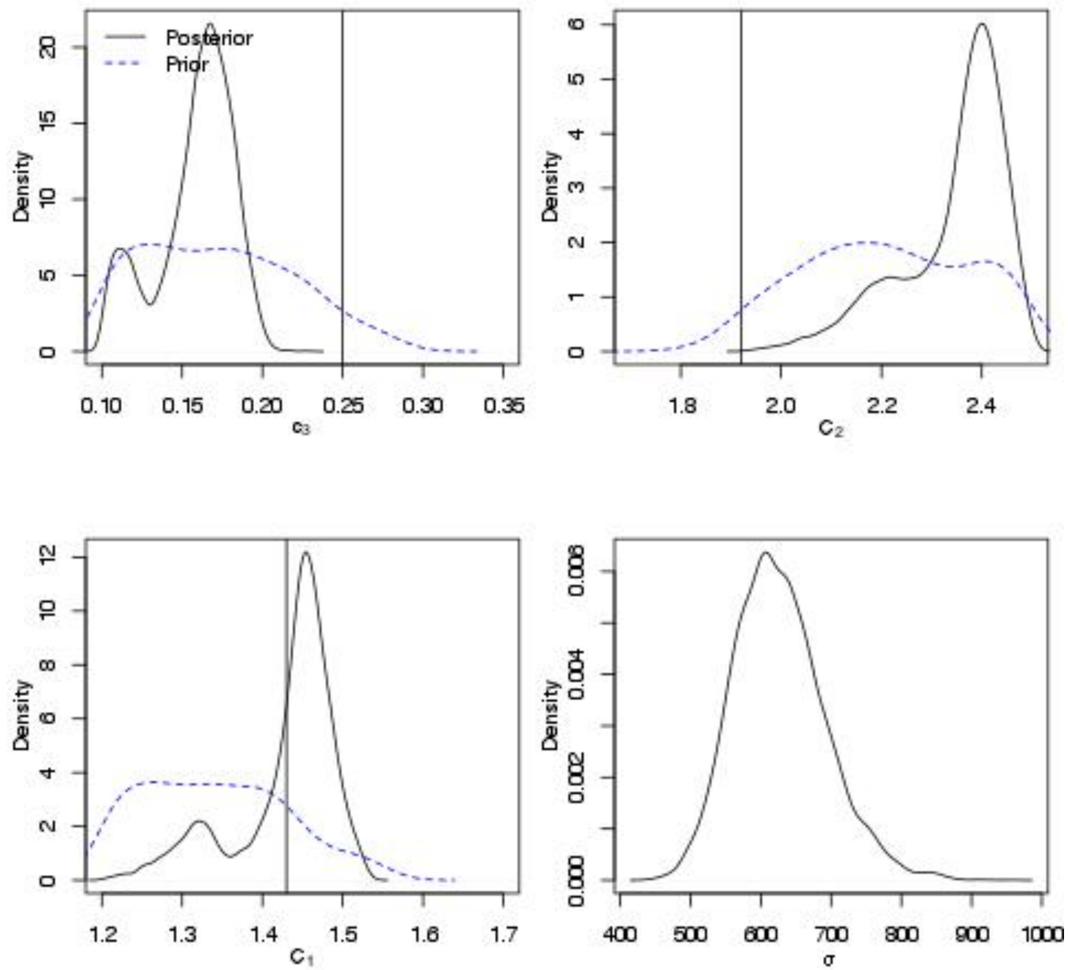
Making the informative prior

- Our emulators are valid only inside $\mathcal{R}$ in the parameter space C
- During the optimization (MCMC) we have to reject parameter combinations outside $\mathcal{R}$ (this is our prior belief $\pi_{prior}(\mathbf{C})$)
  - We define $\zeta(\mathbf{C}) = 1$, for $\mathbf{C}$ in R and $\zeta(\mathbf{C}) = -1$ for $\mathbf{C}$ outside $\mathcal{R}$
  - Then the level set $\zeta(\mathbf{C}) = 0$ is the boundary of $\mathcal{R}$
- The training set of RANS runs is used to populate $\zeta(\mathbf{C})$
- We have to "learn" the discriminating function $\zeta(\mathbf{C}) = 0$
  - We do that using support vector machine (SVM) classifiers



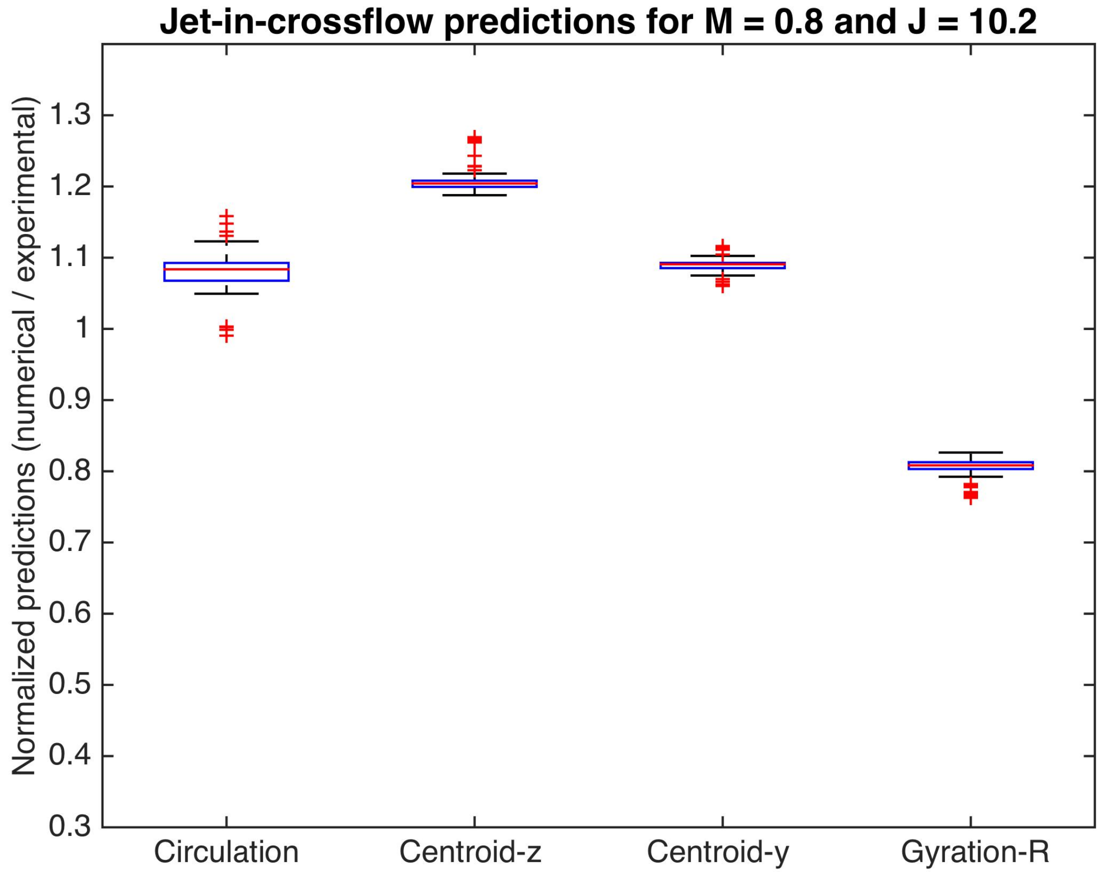
PDFs from the calibration

- About 60,000 MCMC steps to convergence
- Calibrated values of **C** quite different from the ones from literature
  - Vertical lines are the “canonical” values of the parameters
- Next step
  - Draw 100 samples from the posterior distribution and perform RANS simulations
  - Compare with experimental measurements



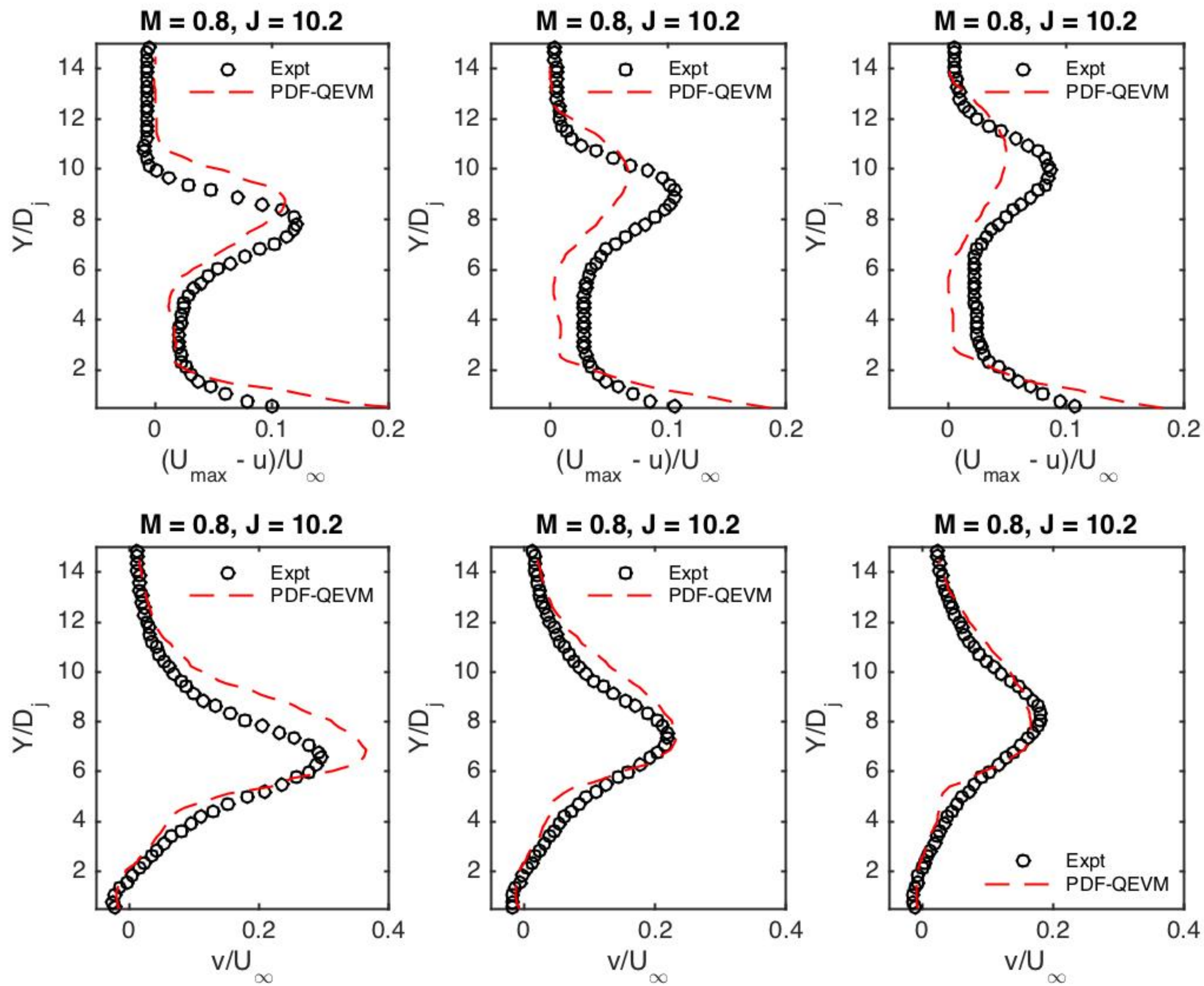
QEVM point vortex metrics

- Compare measured and simulated vorticity fields using the circulation, the centroid and radius of gyration of the vorticity distribution
  - Called the "point vortex metrics"
- Comparable results using existing LEVM models have 20%-70% errors



QEVM PPT predictions on midplane

- Use the 100 RANS simulations to obtain velocity field on the mid-plane
- Compare experimental and simulated predictions

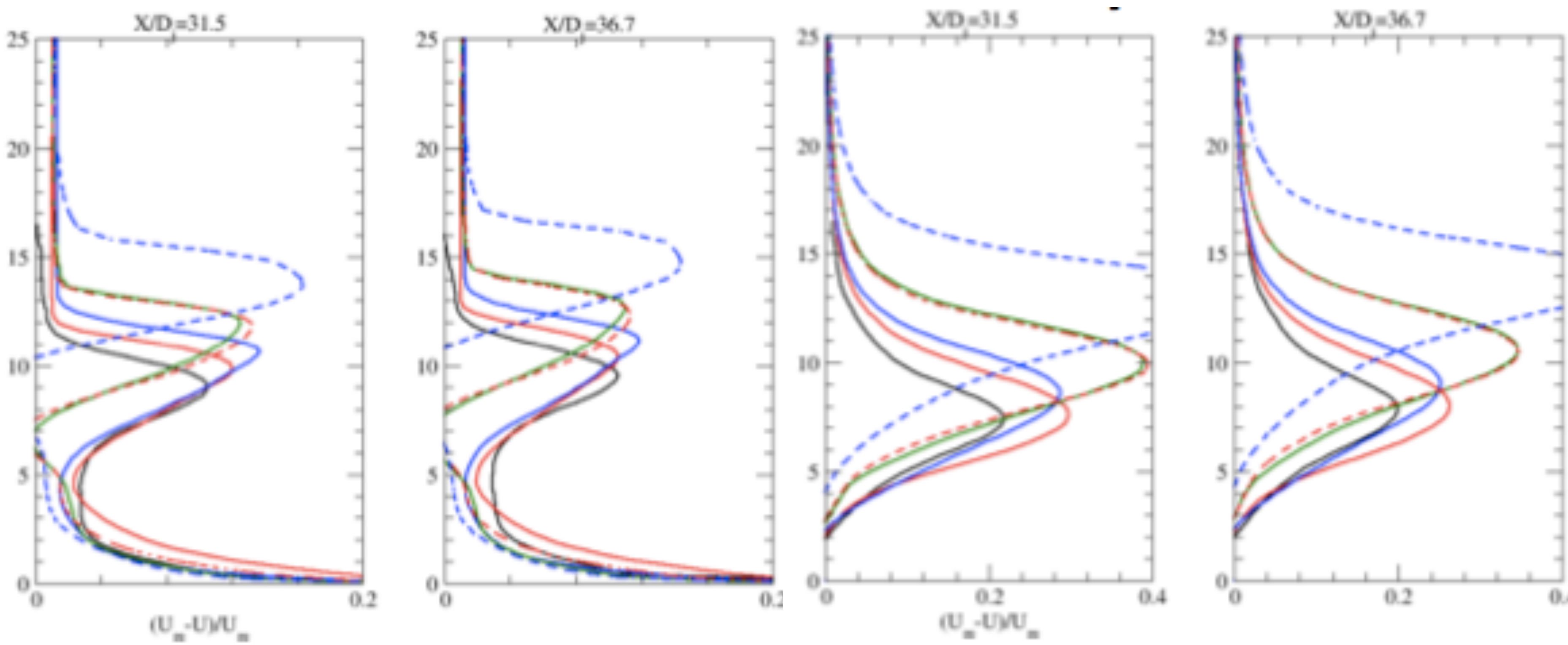


Conclusions

- We are beginning to "fix-up" engineering models with observational data
  - Includes both estimating model parameters and enriching closure models (inferring missing physics in models)
  - Methods are Bayesian; fully probabilistic inference (of parameters, at least)
    - Accommodates uncertainty in estimates due to limited data and shortcomings of the RANS model (model-form error)
- We can tackle rather complicated problems using Bayesian inference
  - Computational costs are immense, but only for generating training data
  - Brittle – we depend on emulators, which can't always be made
  - Can tackle peculiarities of non-physical parameter spaces using informative priors (classifiers)
- Tools and theories: A mixture of statistics and machine learning
  - Bayesian inference, emulators, shrinkage are conventionally statistical
  - Classifiers etc. are purely ML
  - As we scale up and confront large data (simulated flowfields etc.) to infer model-form error, expect MapReduce implementations of these tools

BONEYARD

RANS (k-ω) simulations – midplane results



U-defect

V - velocity

- Experimental results in black
- All models are pretty inaccurate (blue and red lines are the non-symmetric results)

What is MCMC?

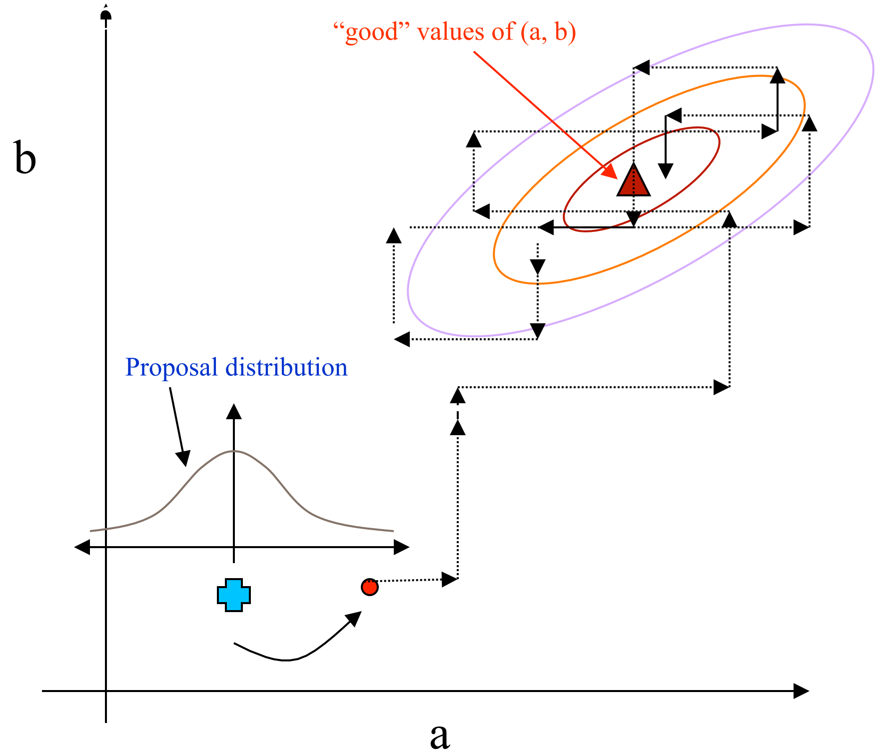
- A way of sampling from an arbitrary distribution
  - The samples, if histogrammed, recover the distribution
- Efficient and adaptive
  - Given a starting point (1 sample), the MCMC chain will sequentially find the peaks and valleys in the distribution and sample proportionally
- Ergodic
  - Guaranteed that samples will be taken from the entire range of the distribution
- Drawback
  - Generating each sample requires one to evaluate the expression for the density π
  - Not a good idea if π involves evaluating a computationally expensive model

An example, using MCMC

- Given: (Y^{obs}, X), a bunch of n observations
- Believed: $y = ax + b$
- Model: $y_i^{obs} = ax_i + b_i + \varepsilon_i$, $\varepsilon \sim \mathcal{N}(0, \sigma)$
- We also know a range where a, b and σ might lie
  - i.e. we will use uniform distributions as prior beliefs for a, b, σ
- For a given value of (a, b, σ), compute "error" $\varepsilon_i = y_i^{obs} - (ax_i + b_i)$
  - Probability of the set (a, b, σ) = $\Pi \exp(- \varepsilon_i^2/\sigma^2)$
- Solution: π (a, b, σ | Y^{obs}, X) = $\Pi \exp(- \varepsilon_i^2/\sigma^2)$ * (bunch of uniform priors)
- Solution method:
  - Sample from π (a, b, σ | Y^{obs}, X) using MCMC; save them
  - Generate a "3D histogram" from the samples to determine which region in the (a, b, σ) space gives best fit
  - Histogram values of a, b and σ, to get individual PDFs for them
  - Estimation of model parameters, with confidence intervals!

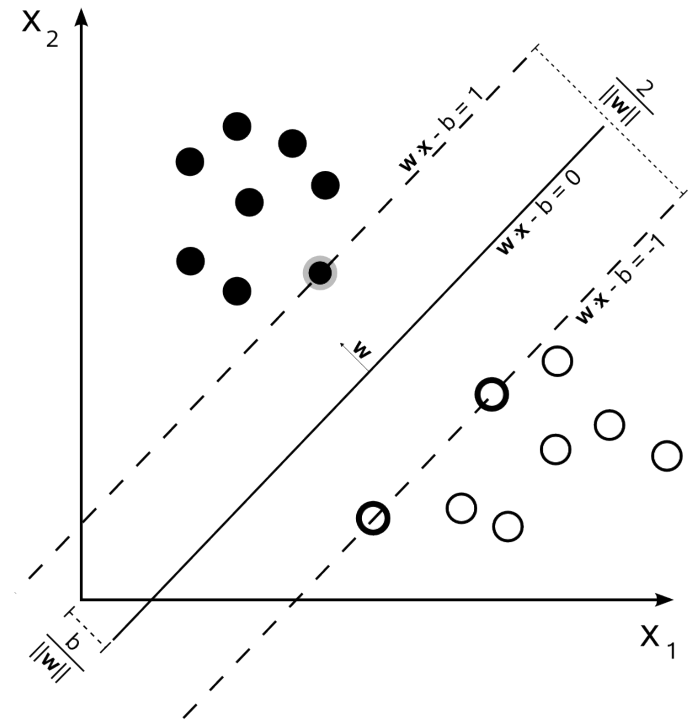
MCMC, pictorially

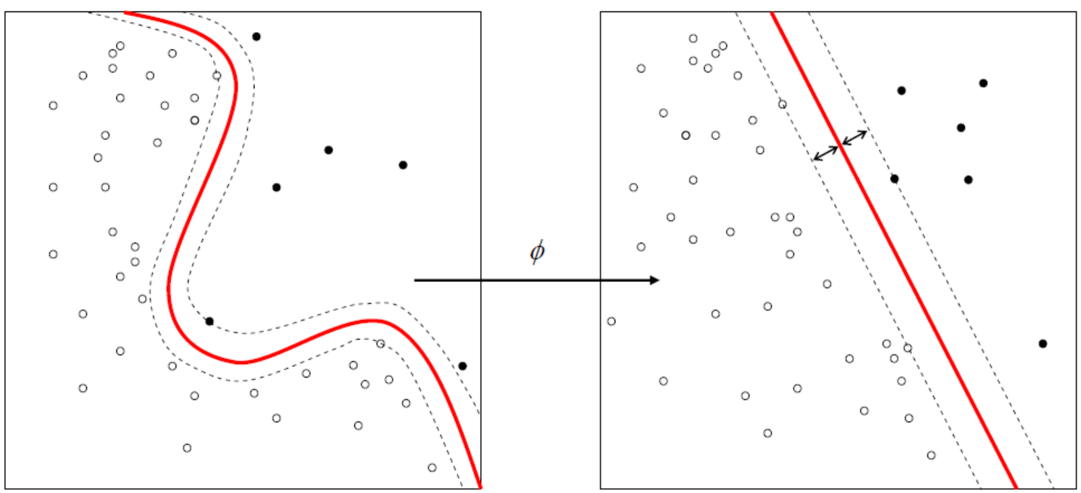
- Choose a starting point, $P^n = (a_{curr}, b_{curr})$
- Propose a new a, $a_{prop} \sim \mathcal{N}(a_{curr}, \sigma_a)$
- Evaluate $\pi (a_{prop}, b_{curr} \mid \ldots) / \pi (a_{curr}, b_{curr} \mid \ldots) = m$
- Accept a_{prop} (i.e. $a_{curr} \leftarrow a_{prop}$) with probability min(1, m)
- Repeat with b
- Loop over till you have enough samples



What is a SVM classifier?

- Given a binary function $y = f(\mathbf{x})$ as a set of points $(y_i, \mathbf{x}_i)$, $y_i = (0, 1)$
  - Find the hyperplane $y + Ax = 0$ that separates the x-space into $y = 0$ and $y = 1$ parts
- Posed as an optimization problem that maximizes the margin





- In case of a curved discriminator, need a transformation first
  - Achieved using kernels
  - We use a cubic kernel