An Introduction to Active Learning

David Cohn
Justsystem Pittsburgh Research Center

- DISCLAIMER: This is a tutorial. There will be no...
  - Gigabyte networks
  - Massive robotic machines
  - Japanese pop stars
- But...
  - you will have the opportunity to shoot the speaker halfway through the talk

A roadmap of today's talk

- Introduction to machine learning
  - what, why and how
- Introduction to active learning
  - what, why and how
- A few examples
  - a radioactive Easter egg hunt
  - robot Tai Chi
  - Gutenberg's nightmare
- The wild blue yonder
  - Active learning "on a budget"
  - What else can we do with this approach?

Machine learning - what and why

- We like to have machines make decisions for us
  - when we don’t have time to - *flight control*
  - when we don’t have attention span to - *large-scale scheduling*
  - when we aren’t available to - *autonomous vehicles*
  - when we just don’t want to - *information filtering*
- Making decision requires evaluating its consequences
- Evaluating consequences may require machine to estimate unknowns or predict future

Machine learning - how to face the unknown?

- Deductive inference - logical conclusions
  - begin with a set of general rules
    - bird(x) $\Rightarrow$ can_fly(x), fish(x) $\Rightarrow$ can_swim(x)
  - follow logical consequences of rules, *deduce* that a specific conclusion is valid:
    - bird(Opus) $\Rightarrow$ can_fly(Opus)
- Inductive inference - the best guess we can make
  - begin with a set of specific examples
    - can_fly(Polly), bird(Polly), can_fly(Albert), bird(Albert), ~can_fly(Flipper), ~bird(Flipper)
  - *induce* a general rule that explains examples: bird(x) $\Leftrightarrow$ can_fly(x)
  - use the rule to deduce new specific conclusions:
    - bird(Opus) $\Rightarrow$ can_fly(Opus)

Machine learning - how to face the unknown?

- If we have a complete rule base, deductive inference is more powerful
  - can prove that our prediction/estimate is correct
- More frequently, don’t have all the information needed for deductive inference
  - *“Should I push the big red button now?”*
  - *“Should I buy 5000 shares of WidgetTech stock?”*
  - *“Is this email from my manager important?”*
  - *“Is that ‘Chocolate Eggplant Surprise’ actually edible?”*
- In these situations, resort to inductive inference

Prediction/estimation with inductive inference

- All sorts of applications require estimating unknowns
  - medical diagnosis: symptoms $\Rightarrow$ disease
  - making oodles of money: market features $\Rightarrow$ tomorrow's price
  - scheduling: job properties $\Rightarrow$ completion time
  - robotic control: motor torque $\Rightarrow$ arm velocity
    - more generally: state $\times$ action $\Rightarrow$ new state
- Make use of whatever information we've got
  - may have complete model, but need to fill in unknown parameters
  - may have partial model - know ordering of relations
  - may know what relevant features are
  - may have nothing but a wild guess

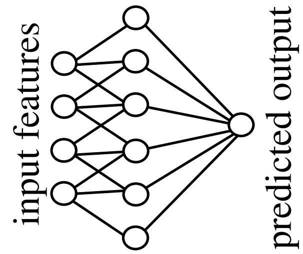
How to predict/estimate

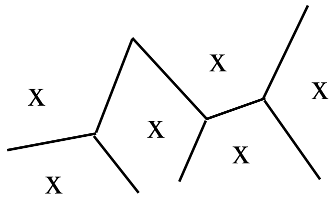
- Need two things for inductive inference:
  - 1) Data - examples of the relation we want to estimate
  - 2) Some means of interpolating/extrapolating data to new values

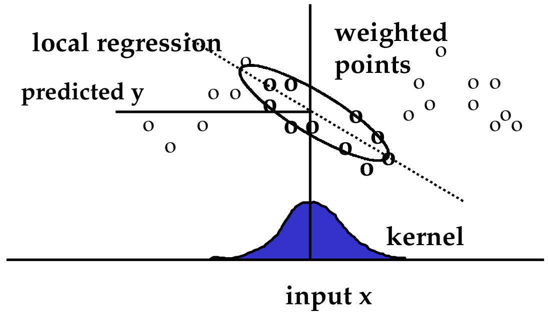
- Focus on (2) for the moment

How to interpolate/extrapolate data

- Parametric models
  - structural models
  - linear/nonlinear regression
  - neural networks
- Non-parametric models
  - k-nearest neighbors
- The weird continuum between
  - locally-weighted regression
  - support vector machines

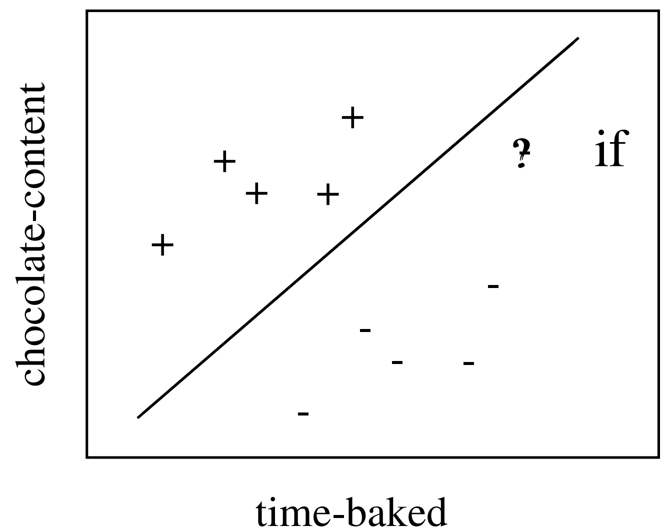






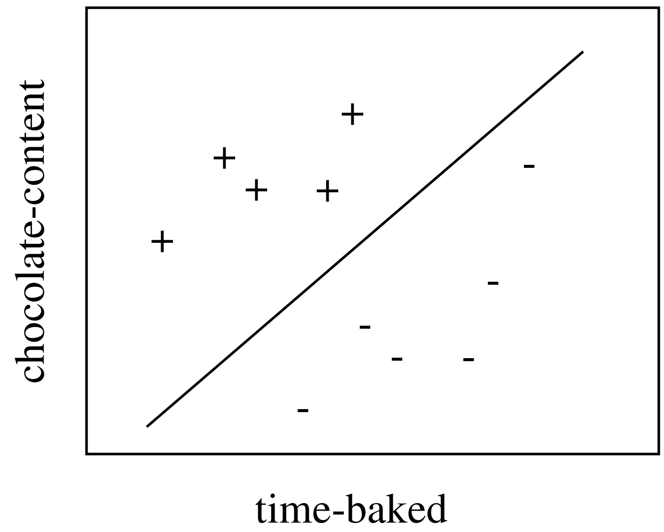
A machine learning example

- Want to build “dessert classifier”
  - predict whether dessert will be edible
- Gather data set of desserts
  - record input features “time-baked”, “chocolate-content”, and output feature “is-edible”
  - use a simple linear classifier
  - perceptron algorithm, many others will find a separating line if one exists



Machine learning - the loss function

- Why place line where we did?
  - “best” decision is one that minimizes loss
  - loss function(al) maps from prediction rule to penalty



- Some common loss functions
  - **MSE** - expected squared error of predictor on future examples
  - **accuracy -** probability that future example will be classified “incorrectly”
  - **entropy** - uncertainty in model parameters
  - **variance** - uncertainty in model outputs

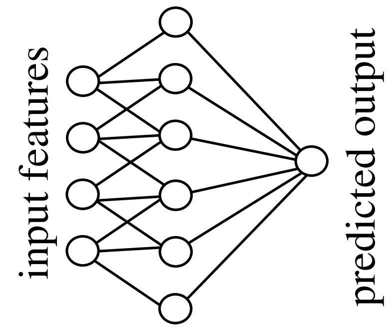
Machine learning - using the loss function

- Machine learning in **three** easy steps
  - 1) Figure out what loss function is for your problem
  - 2) Figure out how to estimate expected loss
  - 3) Find a model that minimizes it

- Huge gobs of time and effort expended on each of these three steps

Machine learning - the typical setup

- Assume known architecture will be used
  - e.g. a neural network
- Assume training set of examples T drawn at random from unknown source S
- Assume loss function
  - e.g. MSE on future examples from S
  - estimate loss via MSE on T
- Find neural network parameters that minimize MSE on T, subject to smoothing and validation conditions




```
T = [(x1,x2,x3,x4 -> y),
     (x1,x2,x3,x4 -> y),
     (x1,x2,x3,x4 -> y),
     ...
     (x1,x2,x3,x4 -> y)]
```

Active learning - what and why

- Goodness of x $\Rightarrow$ y map depends on having

  1) good data to interpolate/extrapolate

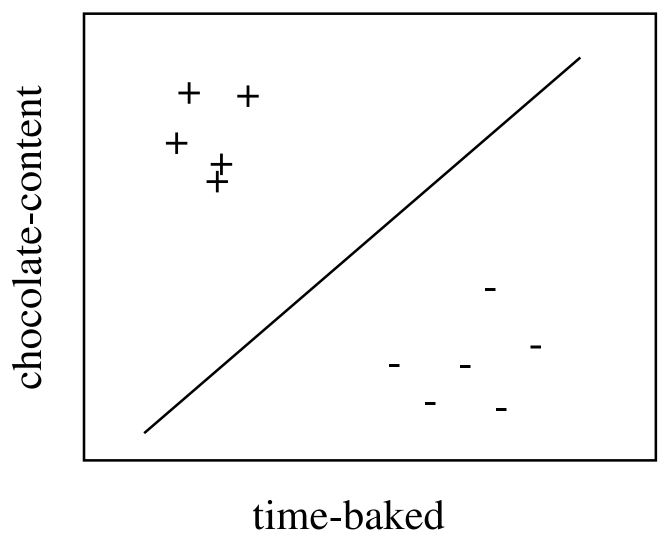
  2) good method of interpolating/extrapolating

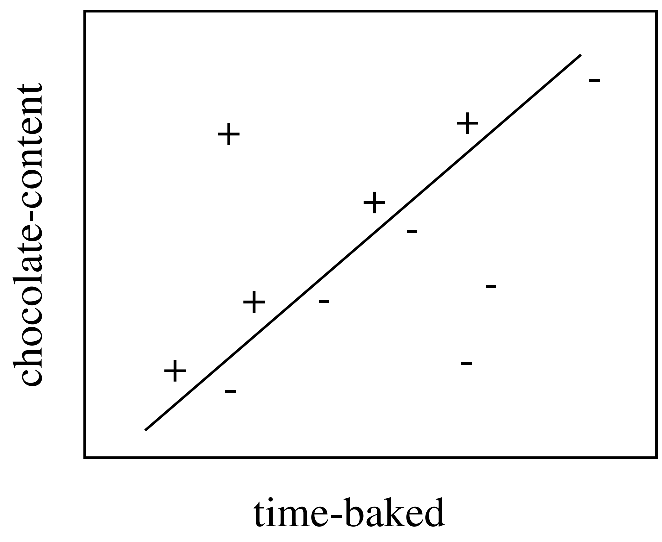
- Machine learning focuses on (2) at the expense of (1)
  - sometimes (1) is out of our hands
    - x-rays, stock market, datamining...
  - sometimes it isn't
    - robotics, vision, information retrieval...

- Active Learning definition: *Learning in which the learner exerts influence over the data upon which it will be trained*
  - Can apply to control, estimation and optimization
  - here, focus on estimation/prediction

Active learning - not all data are created equal

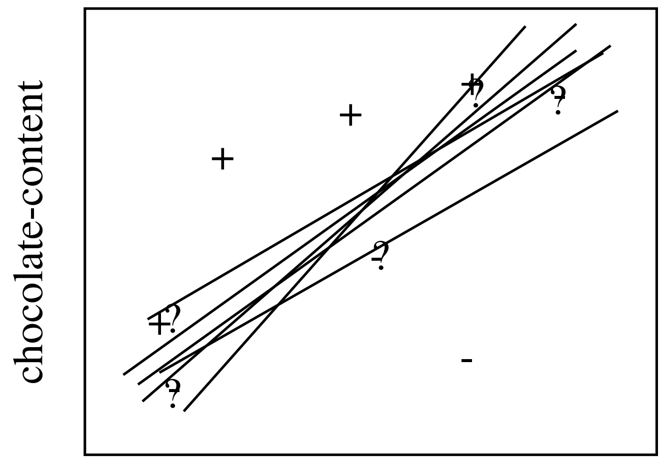
- Depending on model, some data sets will be much better than others
- What data set is best for a model usually cannot be determined a priori
  - must be inferred as you go





An active learning example

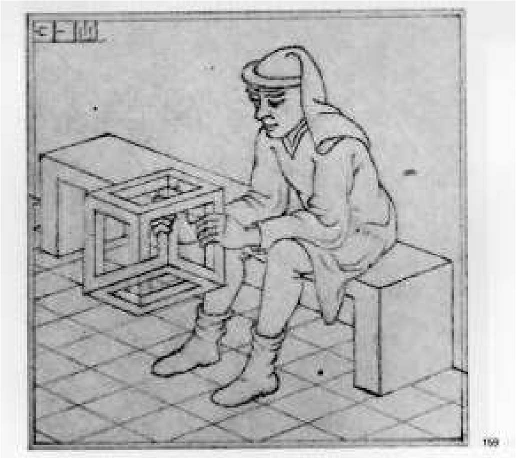
- Want to build active “dessert predictor”
  - predict whether dessert will be edible
- ~~Gather data set of desserts~~
- ***Bake*** a set of desserts, selecting input values that will help us nail down the unknowns in our model



Active learning - why bother?

- Computational costs - selecting data helps us find solutions *faster*
  - in some cases, learning only from given examples is NP-complete, while active learning admits polynomial (or even linear!) time solutions (Angluin, Baum, Cohn)

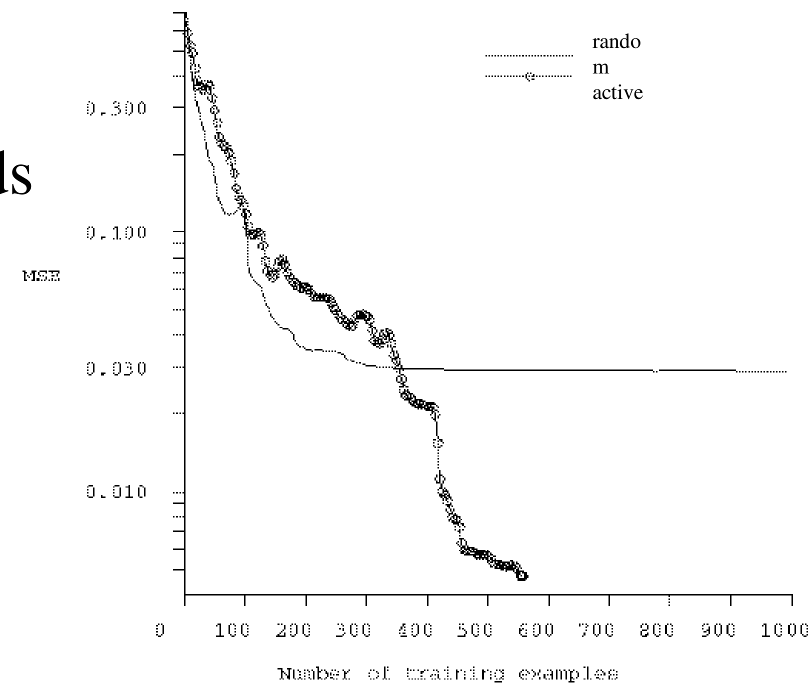
- Example: active vision - having the "right" viewpoint can greatly simplify computation of structure



Active learning - why bother?

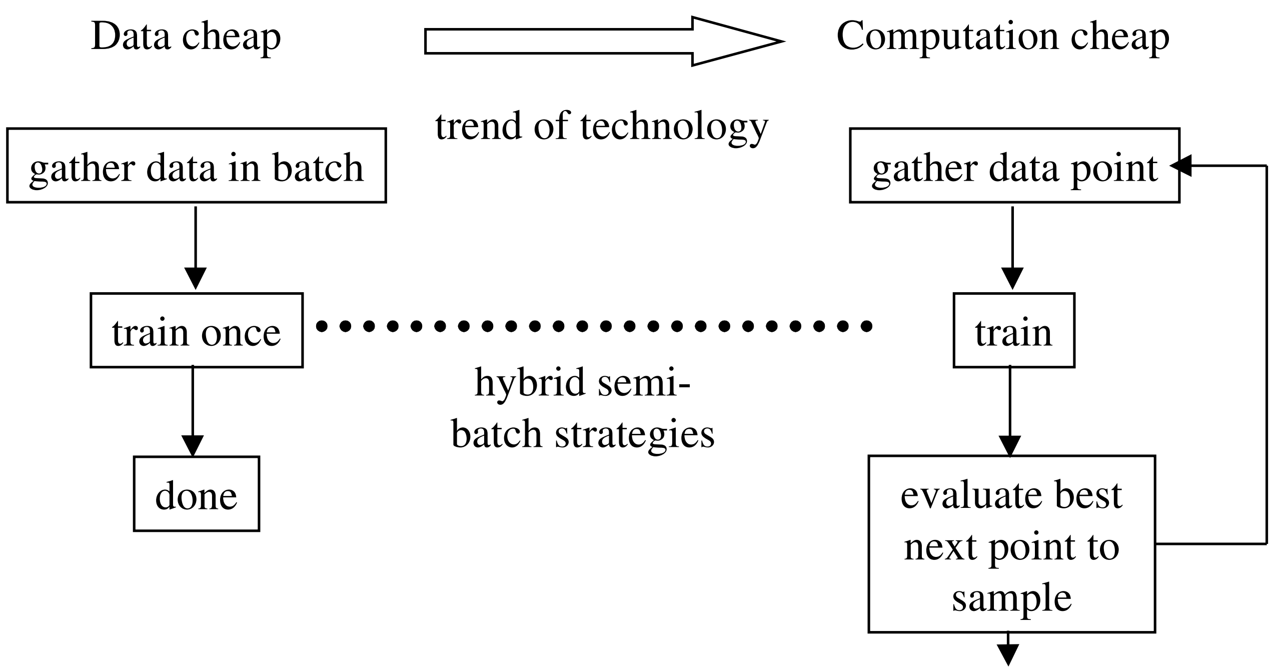
- Data costs - selecting data helps us find *better* solutions
  - in some cases, learning from given examples has a polynomial (or flatter) learning curve, while active learning has exponential learning curve (Blumer et al., Haussler, Cohn & Tesauro)

- Example: learning dynamics - exploring the state space succeeds where random flailing fails



When do we want to do active learning?

- Depends on what our costs are
  - trying to save physical resource?
  - trying to save time? computation?



Active learning in history

- Early mathematical applications
  - given Cartesian coordinates of a target
  - predict angle and azimuth required to shoot it
    - have basic but incomplete Newtonian model that needs tuning
- Process optimization (1950s)
  - George Box - "Evolutionary Operation"
  - explores operating modes in process to hillclimb on yield
- Medicine, Agriculture - optimal experiment design
  - breeding a disease-resistant variety of crop
  - devising a treatment or vaccine
  - generally involve designing batches of experiments

Siblings to active learning

- Persistent excitation - control theory
  - goal is to maintain (near) optimal control of a system
  - vary from the optimal control signal enough to provide continued information about system's parameters
- Optimization - operations research
  - select data/experiments to learn something about shape of response function
  - only interested in *maximum* of function - not general shape

Active learning for estimation

- Active learning in **five** easy steps
  - 1) Figure out what loss function is
  - 2) Figure out how to estimate loss
  - 3) Estimate effect of a new candidate action/example on loss
  - 4) Choose candidate yielding smallest expected loss
  - 5) Repeat as necessary

A few examples

- Active learning with a parametric model
  - a radioactive Easter egg hunt
- Active learning for prediction confidence
  - robot Tai Chi
- Active learning on a big ugly problem
  - Gutenberg’s nightmare

Active learning with a parametric model

- Locate buried hazardous materials
  - barrels of hazardous waste buried in unmarked locations
  - metal content causes electromagnetic disturbance which can be measured at surface
  - want to localize barrels with minimum number of probes

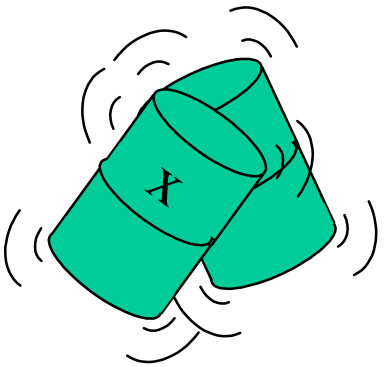
Active learning with a parametric model

- We have a parametric model of disturbances, but individual probes are very noisy
- Given a barrel buried at *(x_0, y_0, z_0)* , mean disturbance a probe location *(x, y, z)* is :

$$h(x,y,z) = \frac{3k(z-z_0)^2}{r^5} - \frac{k}{r^3}$$

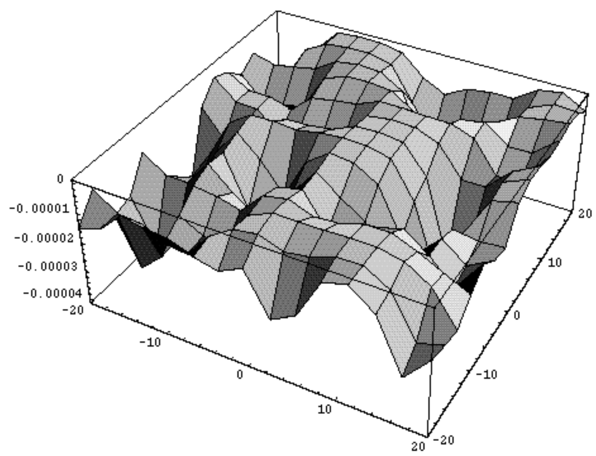
where

$$r = \sqrt{(x-x_0)^2 + (y-y_0)^2 + (z-z_0)^2}$$

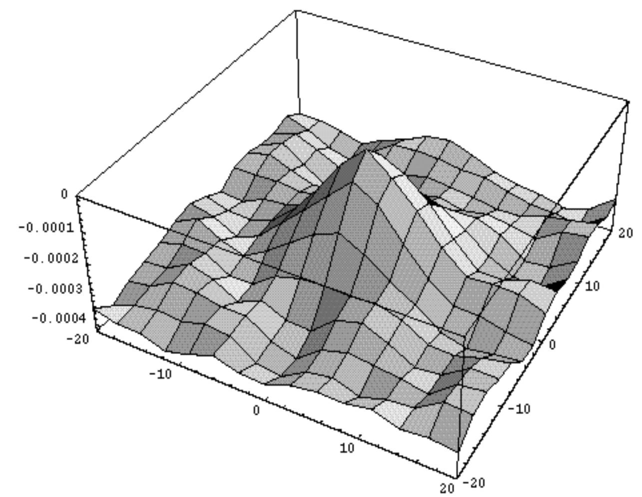


Active learning with a parametric model

- Given data and a noise model, apply Bayes rule and do maximum likelihood estimation of parameters from data:
  - $P(x_0, y_0, z_0 | D)$
  - provides confidence estimate for any hypothesized barrel location (x_0, y_0, z_0)



after 60 random probes



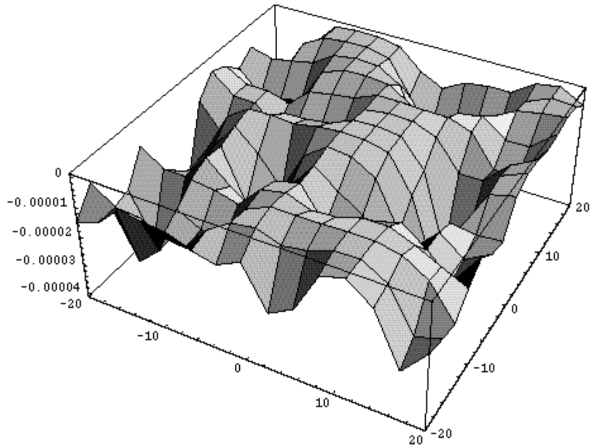
after 1200 random probes

Active learning with a parametric model

- Use current likelihood map to decide where to make next probe
- A few possible strategies:
  - make probes at random - inefficient
  - "the beachcomber" - take next probe at most likely location
  - "the engineer" - follow the "five easy steps" of active learning

Active learning with a parametric model

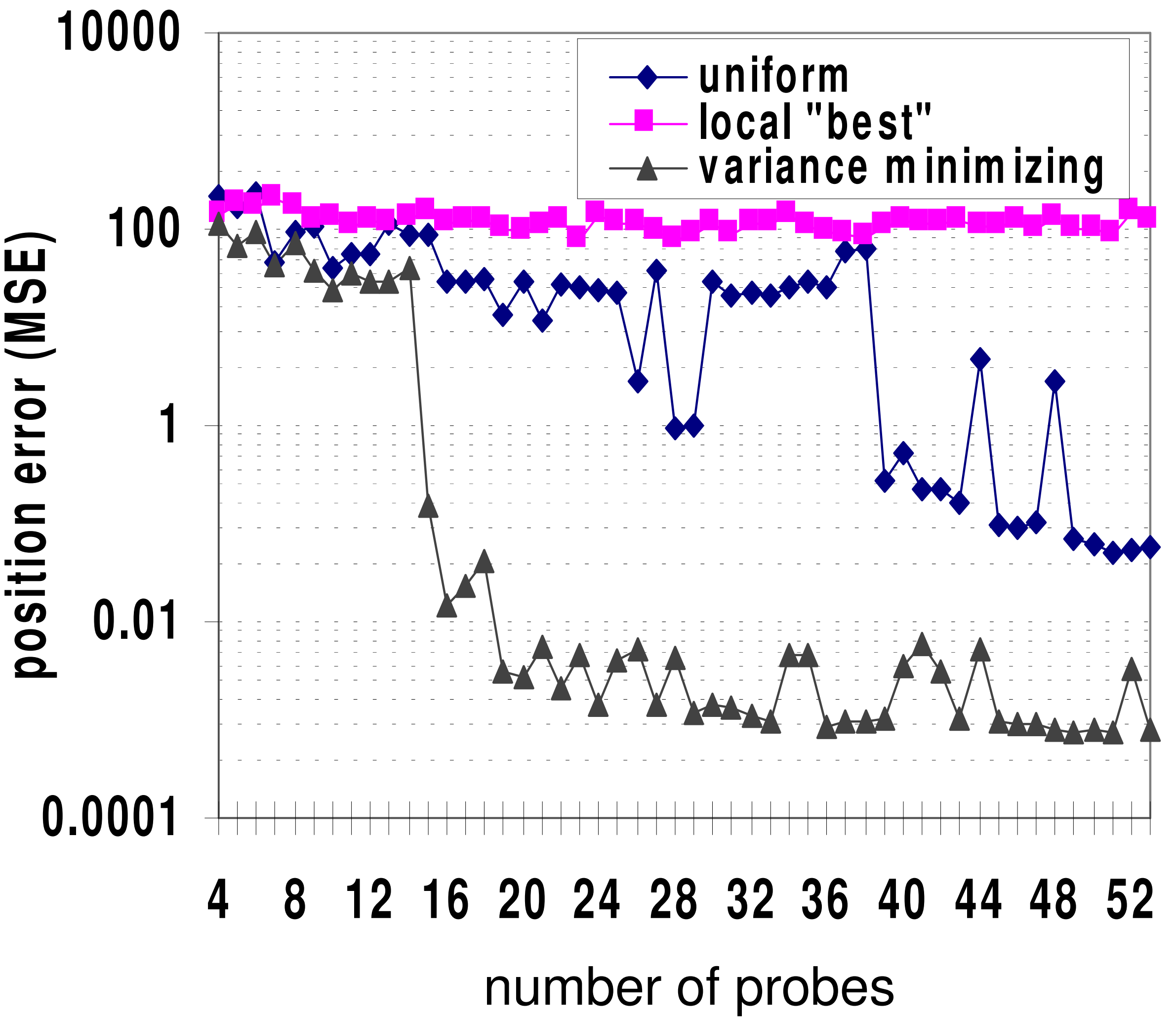
- Five easy steps:
  1) loss function is MSE between our estimates and true location of (x_0, y_0, z_0)
  2) can estimate loss with variance of parameter MLE
  3) estimate effect of new probe at (x', y', z') on MLE
  4) identify (x', y', z') that minimizes variance of MLE
  5) query, and repeat as necessary



Active learning with a parametric model

- How estimate effect of new probe at *(x', y', z')* on MLE?
- If we knew *(h'|x', y', z')* it would be easy
- Estimate *h'* with Bayesian approach
  - if true location of barrel is (x_0, y_0, z_0), can compute distribution $P(h' | x', y', z', D)$ from noise model
  - weight distribution of h' by likelihood of (x_0, y_0, z_0), given current data
  - integrate over all reasonable (x_0, y_0, z_0) to arrive at expected distribution of responses $P(h' | x', y', z')$

Active learning with a parametric model



Active learning for prediction confidence

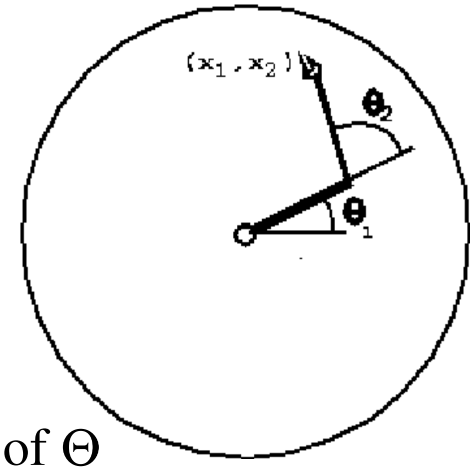
- Frequently, model parameters are a means to an end
  - e.g. in a neural network, parameters are meaningless
  - don't care how confident we are of parameters - we want to be confident of outputs
    - this turns out to be a tad more tricky!
- Output confidence must be integrated over entire domain
  - prediction confidence at any point x straightforward
    - compute analytically, or estimate using Taylor series or Monte Carlo approximations
  - but overall confidence must be integrated for all x of interest
    - requires knowing test distribution

Active learning for prediction confidence

- Need to integrate uncertainty over entire domain
  - requires estimate of test distribution *p(x)*
  - passive learning traditionally uses training set for estimate of *p(x)*
  - *But if we've been choosing the training data.... (oops!)*
- We're still okay if...
  - we can define the test distribution, or
  - we can approximate the test distribution, or
  - have access to a large number of unlabeled examples
- Do Monte Carlo integration over a "reference set"
  - draw unlabeled reference set X_{ref} according to test distribution
  - estimate variance at each point x_{ref} in reference set

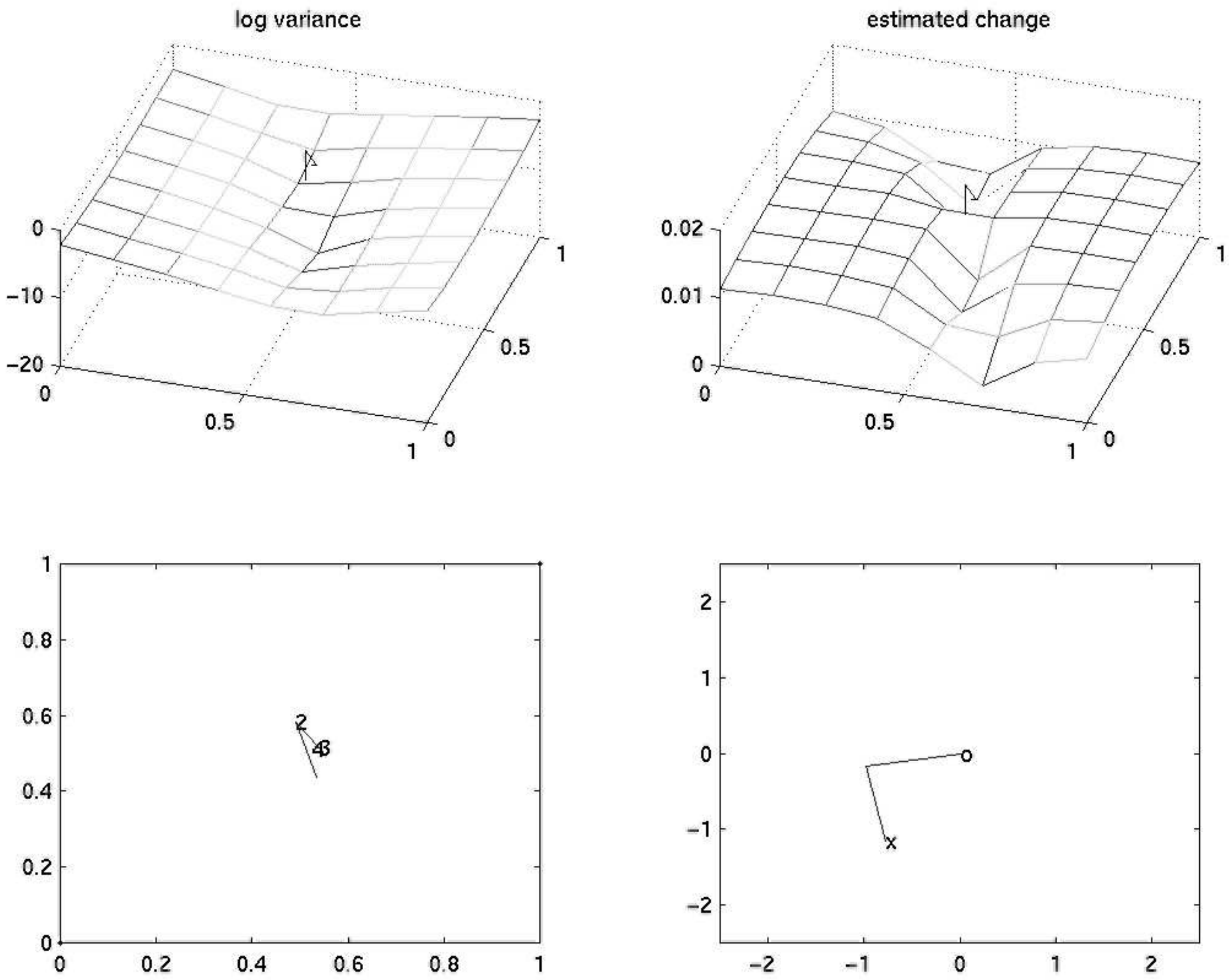
Active learning for prediction confidence

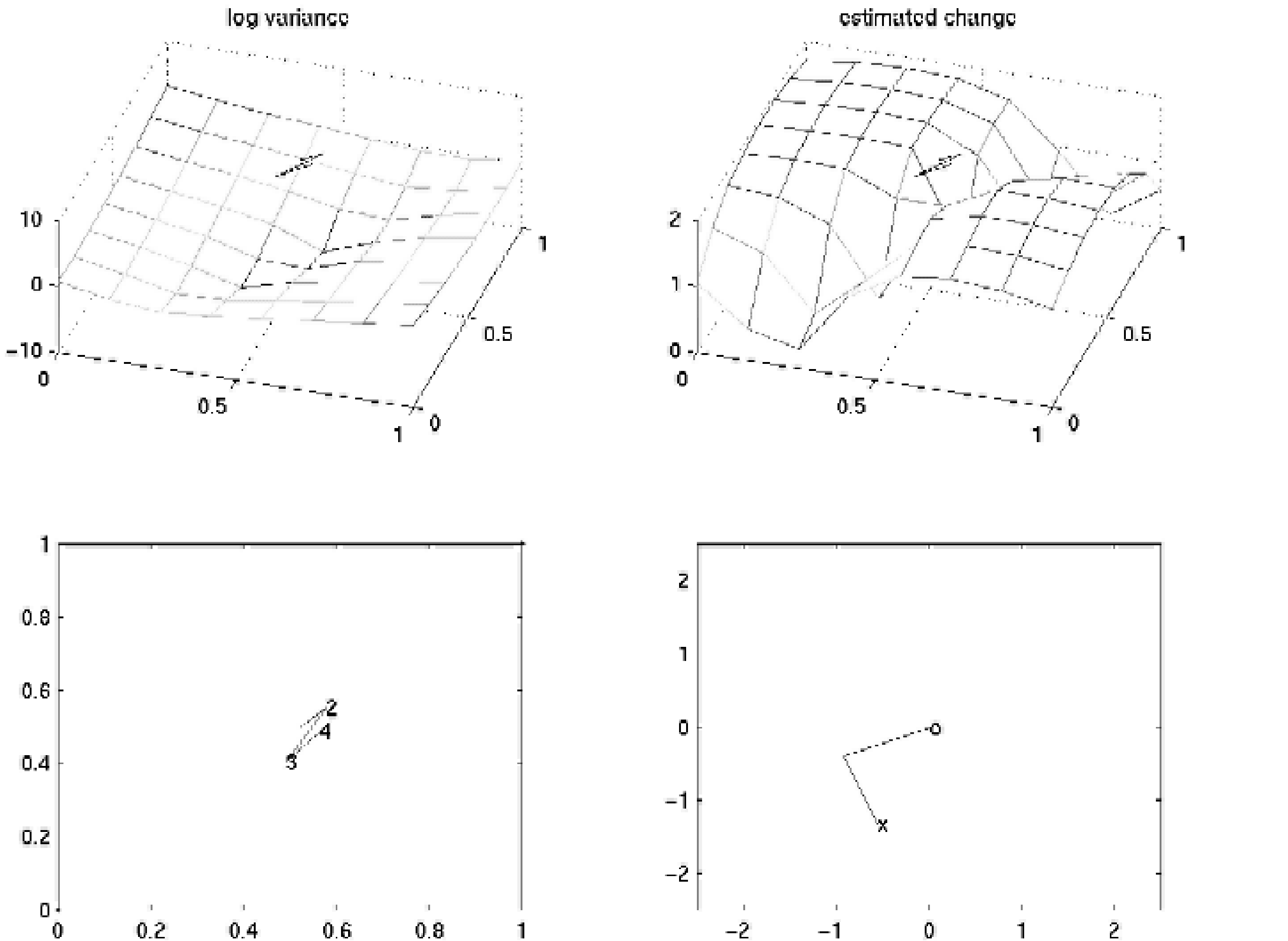
- Learning kinematics of a planar two-joint arm
  - inputs are joint angles Θ_1, Θ_2
  - outputs are Cartesian coordinates x_1, x_2
  - Gaussian noise in angle sensors, effectors produces non-Gaussian noise in Cartesian output space



- Loss function is uniform MSE over Θ_1, Θ_2
- Select successive Θ's to minimize loss
- Two versions of problem
  - stateless: successive queries can be arbitrary values of Θ
  - with state: successive queries must be within r of prior Θ
- Pick locally weighted regression as model architecture

Active learning with LWR- a demo





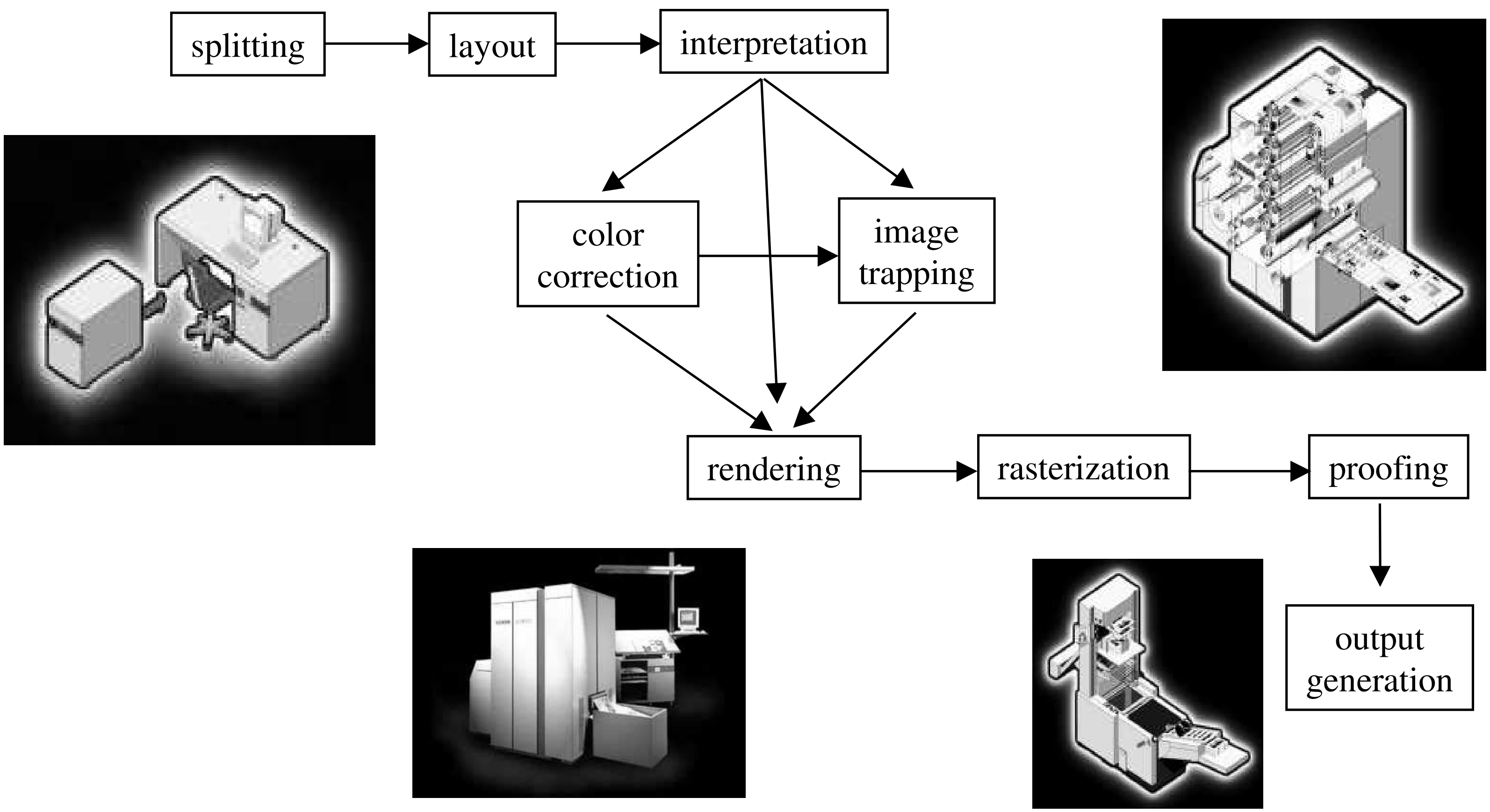
log variance
estimated change
10
0
−10
0
0.5
1
0
0.5
1
2
1
0
0
0.5
1
0
0.5
1
1
0.8
0.6
0.4
0.2
0
0
0.2
0.4
0.6
0.8
1
2
4
3
2
1
0
−1
−2
−2
−1
0
1
2
o
x

Active learning to minimize bias *and* variance

- Maximizing confidence in model parameters and outputs assumes that the model is right
  - but models are almost *never* right!
  - discrepancy shows up as model bias
- Can use many of the same tricks to select data that will minimize bias and variance simultaneously
- Get concomitant improvement in performance

Life in a digital prepress print shop

- Real-time stochastic scheduling, or “Gutenburg’s nightmare”



Life in a digital prepress print shop

- The scale of the problem
  - 50-100 machines
  - 100's of tasks at any given moment
  - machines added, disappearing, changing on day-by-day basis
  - tasks added, disappearing, changing on minute-by-minute basis
- EP2000 - dragging digital prepress out of the 1600's
  - Integrated workflow management/optimization system for DPP
    - cost, deadline requirement determined when job arrives
    - jobs are decomposed into tasks and dependencies
    - resource requirements estimated for each task
    - tasks scheduled, executed

The prediction problem in EP2000

- In order to do scheduling, need to estimate resource requirements for each task
  - example: How long to rasterize this PostScript file on a DCP/32S?

```
%! by HAYAKAWA,Takashi<h-takasi@isea.is.titech.ac.jp>
/p/floor/S/add/A/copy/n/exch/i/index/J/ifelse/r/roll/e/sqrt/H{count 2 idiv exch
repeat}def/q/gt/h/exp/t/and/C/neg/T/dup/Y/pop/d/mul/w/div/s/cvi/R/rlineto{load
def}H/c(j1idj2id42rd)/G(140N7)/Q(31C85d4)/B(V0R0VRVC0R)/K(WCVW)/U(4C577d7)300
T translate/I(3STinTinTinY)/l(993dC99Cc96raN)/k(X&E9!&1!J)/Z(blxC1SdC9n5dh)/j
(43r)/O(Y43d9rE3IaN96r63rvx2dcaN)/z(&93r6IQO2Z4o3AQYaNlxS2w!)/N(3A3Axe1nwc)/W
270 def/L(1i2A00053r45hNvQXz&vUX&UOvQXzFJ!FJ!J)/D(cjS5o32rS4oS3o)/v(6A)/b(7o)
/F(&vGYx4oGbxSd0nq&3IGbxSGY4Ixwca3AlvvUkbQkdbGYx4ofwnw!&vlx2w13wSb8Z4wS!J!)/X
(4I3Ax52r8Ia3A3Ax65rTdCS4iw5o5IxnwTTd32rCST0q&eCST0q&D1!&EYE0!J!&EYEY0!J0q)/V
0.1 def/x(jd5o32rd4odSS)/a(1CD)/E(YYY)/o(1r)/f(nY9wn7wpSps1t1S){[n{( )T 0 4 3 r
put T(/)q{T(9)q{cvn}{s}J}{($)q{[}{]}J}J cvx}forall]cvx def}H K{K{L setgray
moveto B fill}for Y} bind for showpage
```

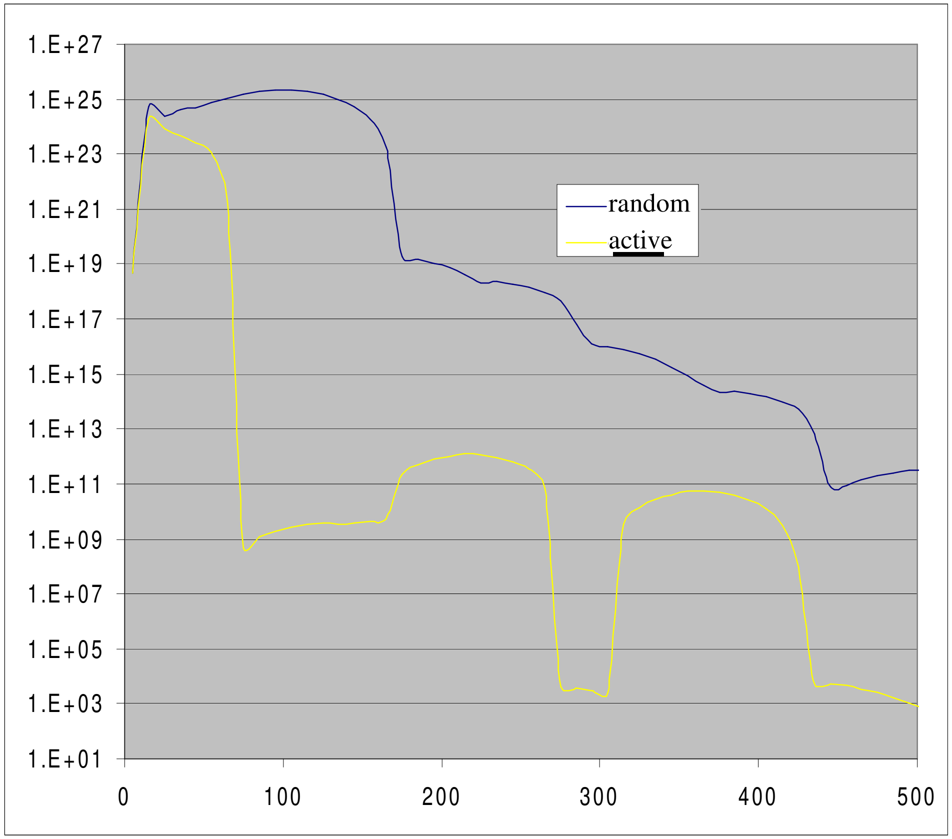
- Estimate time from
  - surface features of input files (length, number of fills, area of fills...)
  - features of the target machine (clock speed, RAM, cache, disk speed)

Resource estimation in EP2000

- Requirements:
  - predict quickly and accurately
  - incorporate new information quickly
- Analytic estimation intractable - so use machine learning
  - detailed simulation model too complex
  - use locally-weighted regression on selected subset of features
- Generating accurate model is time-consuming
  - when a new resource comes online, it must be calibrated
    - how long will task T take on machine M?
    - run a series of test jobs to calibrate predictions
- *The active learning bits: which jobs will calibrate machine most quickly?*

Active learning in EP2000

- Selective sampling
  - hard to generate synthetic jobs to run
  - instead select calibration jobs from a large set of available benchmark tasks



A few places I've pulled the wool over your eyes

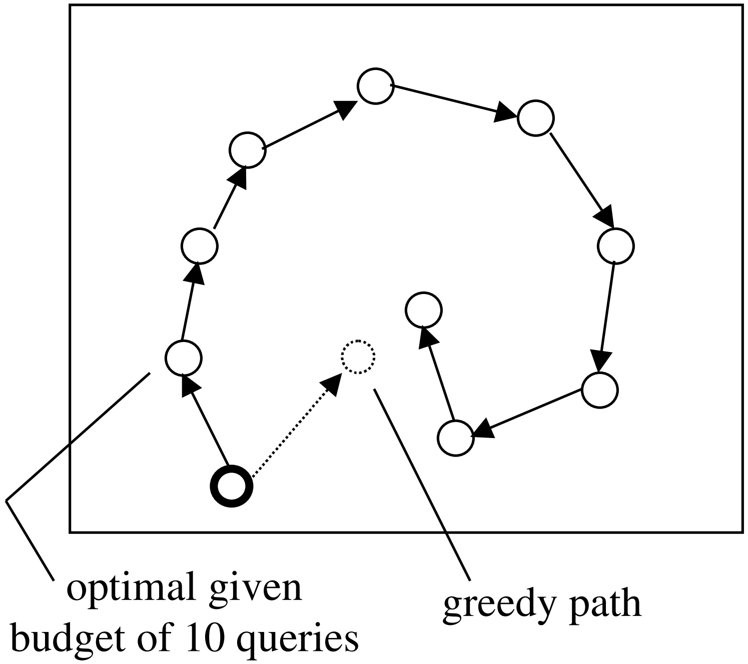
- Computational rationality
  - by thinking about which calibration job to run next, we're spending time thinking to save time running
  - at what point is it better to stop thinking, and just ***do***?

- Just what *is* the loss function for a prediction algorithm whose output is fed to a scheduler?

- "What do I do next?" provides a greedy solution - not a truly optimal one

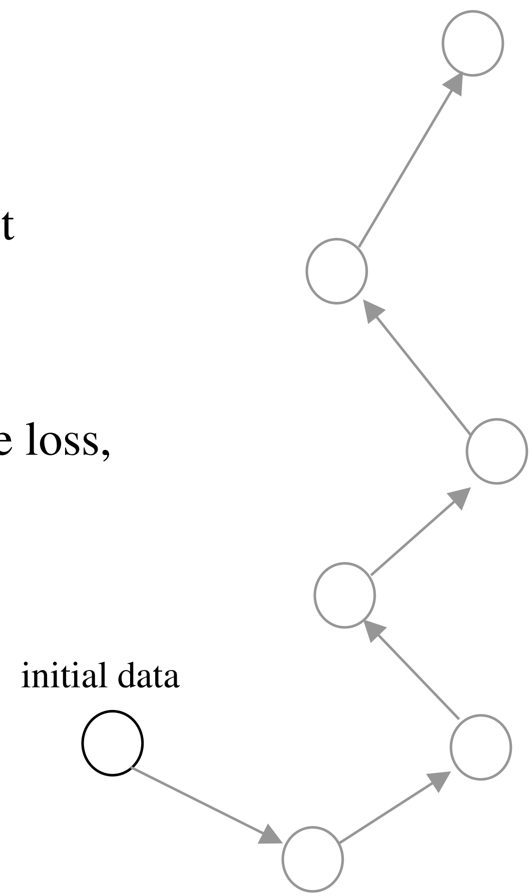
What happens when we have a budget?

- Greedy approach is not optimal
  - Knowing experimental “budget” provides strategic information - how do we want to spend our experiments?
  - Budget may be in terms of
    - sample size - how many experiments?
    - known cost - tradeoff cost/benefit
    - unknown cost - must guess
  - Example: calibrating on a deadline
    - have 24 hours to calibrate machine
    - have large set of calibration files
    - each run takes unknown time
    - select set of files for best calibration before deadline



An algorithm for active learning on a budget

- An EM-like approach:
  1) Build feedforward greedy strategy
     - select best next point to query
     - guess result of query, simulate addition of result
     - iterate
  2) Gauss-Seidel updates
     - iteratively perturb individual points to minimize loss, given estimated effect of other points

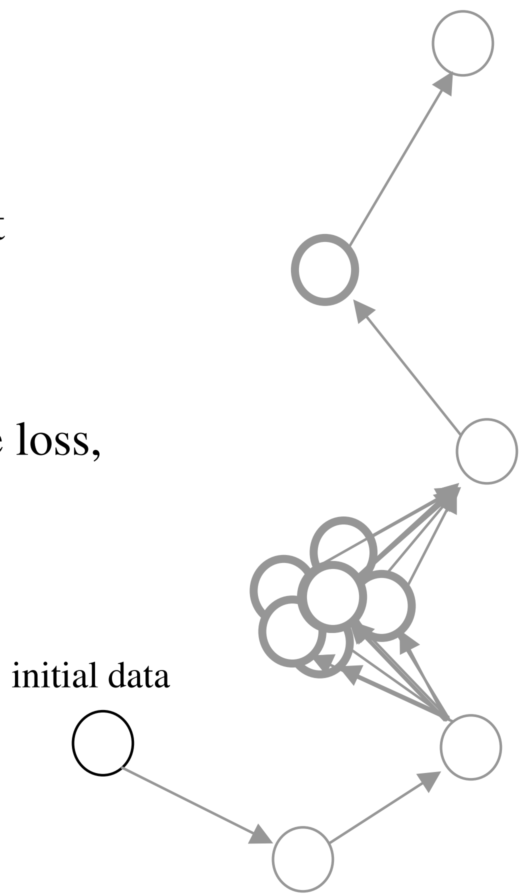


An algorithm for active learning on a budget

- An EM-like approach:

  1) Build feedforward greedy strategy
    - select best next point to query
    - guess result of query, simulate addition of result
    - iterate

  2) Gauss-Seidel updates
    - iteratively perturb individual points to minimize loss, given estimated effect of other points

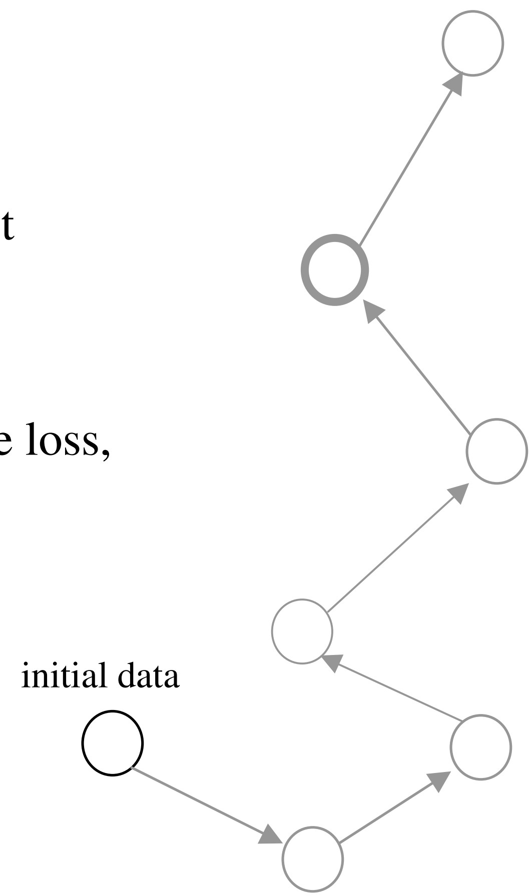


An algorithm for active learning on a budget

- An EM-like approach:
  1) Build feedforward greedy strategy
     - select best next point to query
     - guess result of query, simulate addition of result
     - iterate

  2) Gauss-Seidel updates
     - iteratively perturb individual points to minimize loss, given estimated effect of other points

- Huge increase in computational cost
  - greedy method requires *O(n)* optimizations
  - iterative method requires $O(kn^2)$
    - *k* is number of iterative perturbations

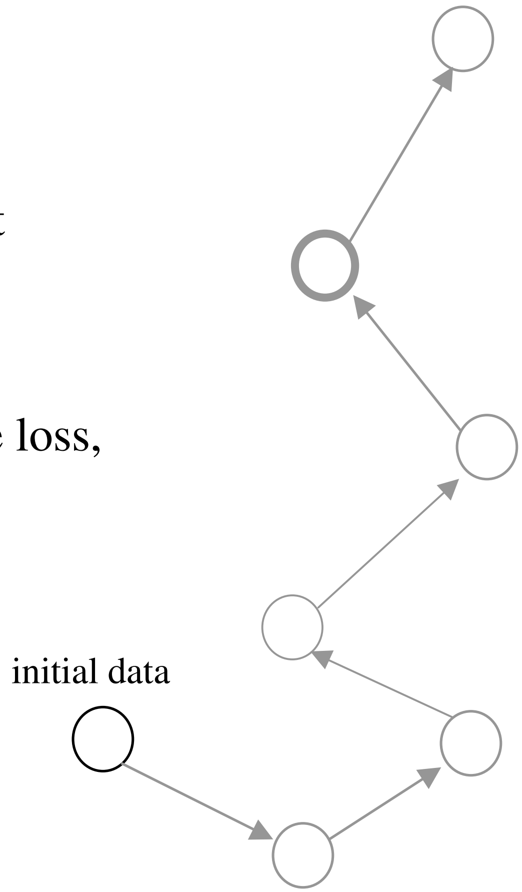


An algorithm for active learning on a budget

- An EM-like approach:
  1) Build feedforward greedy strategy
    - select best next point to query
    - guess result of query, simulate addition of result
    - iterate

  2) Gauss-Seidel updates
    - iteratively perturb individual points to minimize loss, given estimated effect of other points

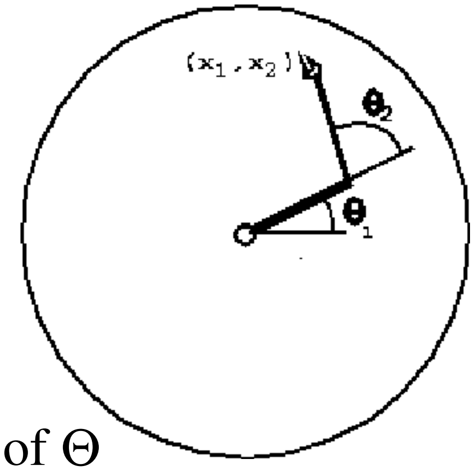
- Huge increase in computational cost
  - greedy method requires $O(n)$ optimizations
  - iterative method requires $O(kn^2)$
    - k is number of iterative perturbations



- Question: does computational cost outweigh benefit?

Active learning on a budget

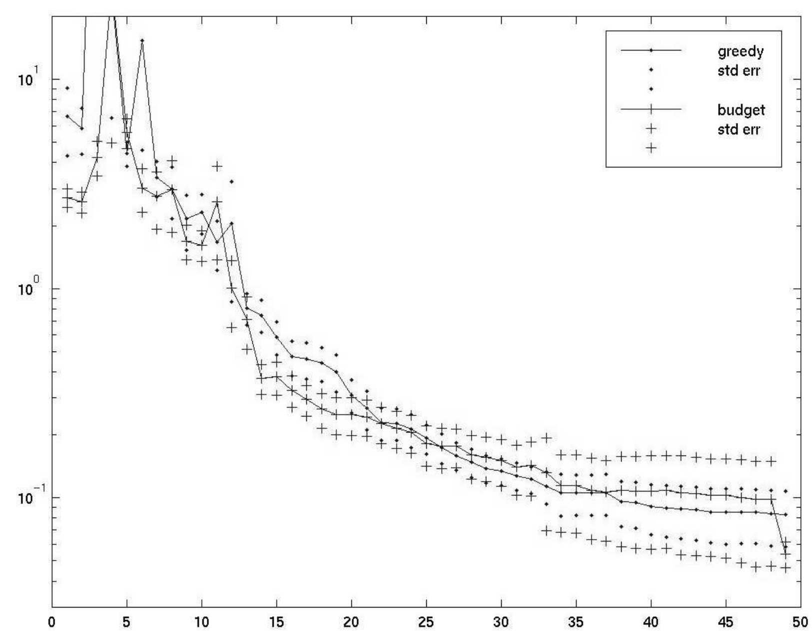
- Learning kinematics of a planar two-joint arm
  - inputs are joint angles Θ_1, Θ_2
  - outputs are Cartesian coordinates x_1, x_2
  - Gaussian noise in angle sensors, effectors produces non-Gaussian noise in Cartesian output space



- Loss function is uniform MSE over Θ_1, Θ_2
- Select successive Θ's to minimize loss
- Two versions of problem
  - stateless: successive queries can be arbitrary values of Θ
  - with state: successive queries must be within r of prior Θ
- Pick locally weighted regression as model architecture

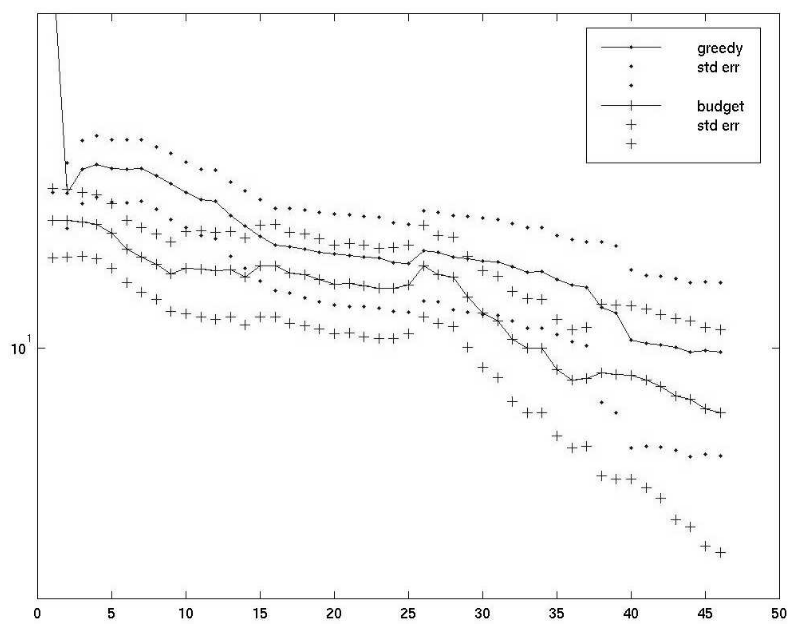
Active learning on a budget

- Stateless domain
  - computationally *very* expensive
  - ~1-2 hours for each example
  - very little improvement over greedy learning



Active learning on a budget

- Domain with state
  - computationally *very* expensive
    - ~1-2 hours for each example
  - significant improvement over greedy learning, but high variance
    - sometimes performs very poorly
    - algorithm is clearly not achieving full potential of domain



Great - where else can this stuff be used?

- Document classification and filtering
  - learn model of what sort of articles I like to see
  - learn how to file my email into the right mailboxes
  - identify what I'm looking for
  - *"Don't pester me - only ask me important, useful questions"*
    - can eliminate > 90% of queries
- Robotics
  - What action will give us the most information about environment?
    - select camera positions to support/refute hypotheses about scene structure
    - select torques/contact angles of robotic effector to provide information about unknown material
    - select course/heading to explore uncharted terrain

Discussion

- Machine learning - what have we learned?
  - Sometimes it's a darned good idea
- Active learning - what have we learned?
  - carefully selecting training examples can be worthwhile
  - "bootstrapping" off of model estimates can work
  - sometimes, greed is good
- Where do we go from here?
  - more efficient sequential query strategies
    - borrow from planning community
  - computationally rational adaptive systems - when is optimality worth the extra effort?
    - borrow from work on 'value of information'