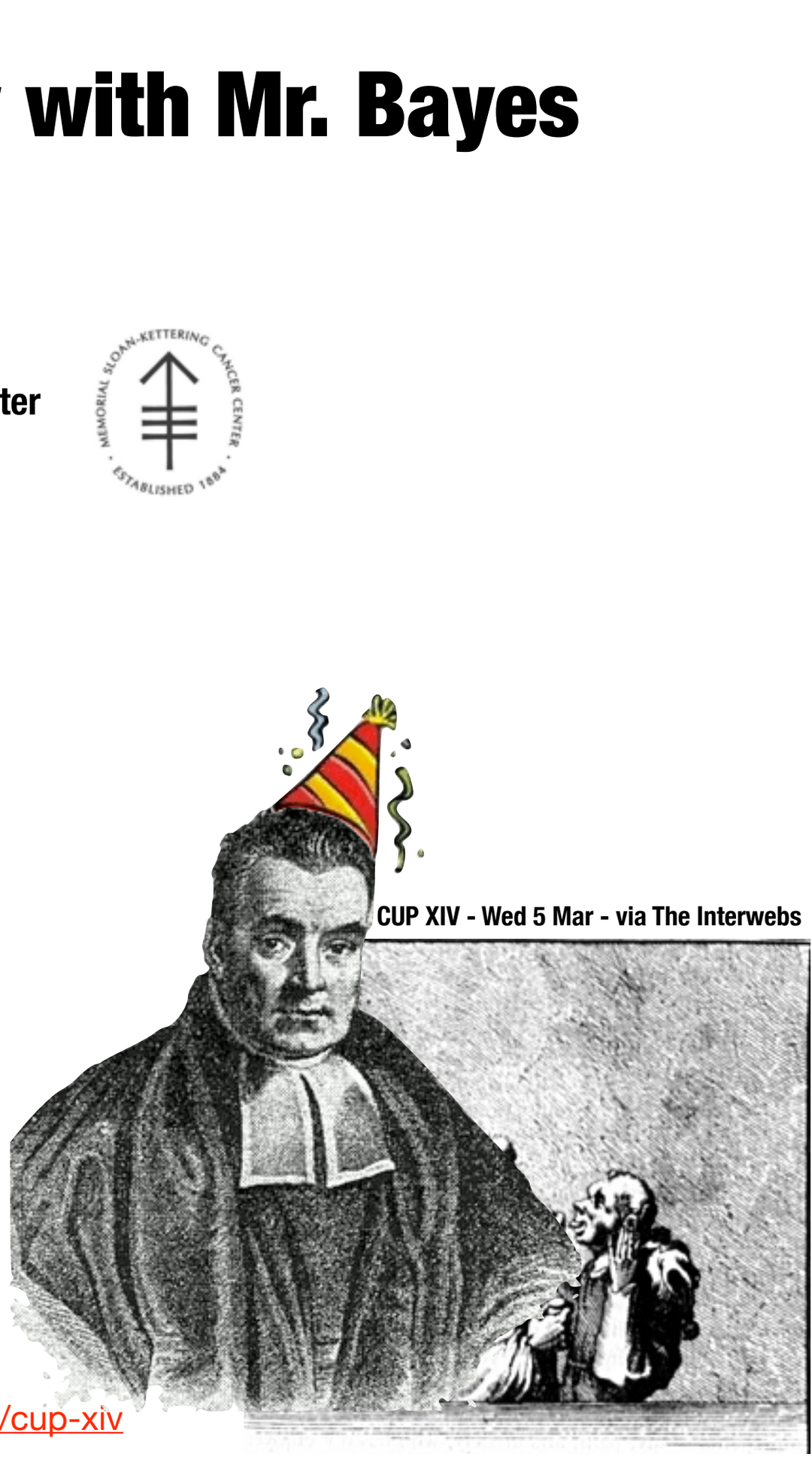
Story time and puppet show with Mr. Bayes

Data, Models, and Decisions

John D. Chodera
Computational Biology Program, Memorial Sloan-Kettering Cancer Center
http://www.choderalab.org

Tweet your questions to @jchodera



Slides, references, and IPython tutorials available at http://choderalab.org/cup-xiv


iRobot
Roomba
SPOT
CLEAN

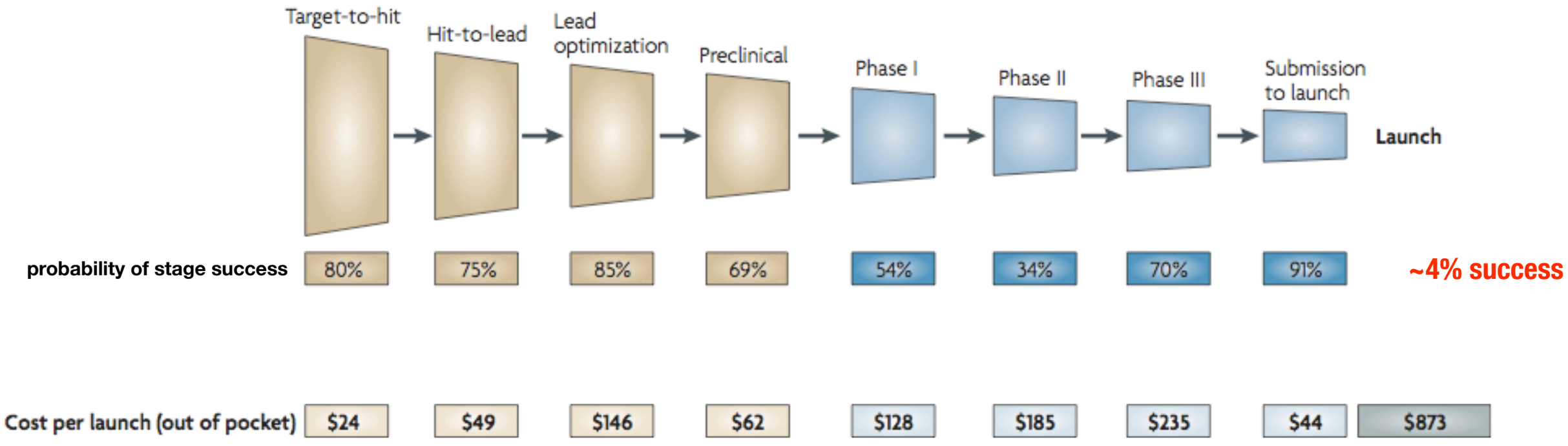
We are all here because we desperately hope modeling can improve the dismal success rates in drug discovery

---

Total pharma research spending has doubled to $65.3B in last decade

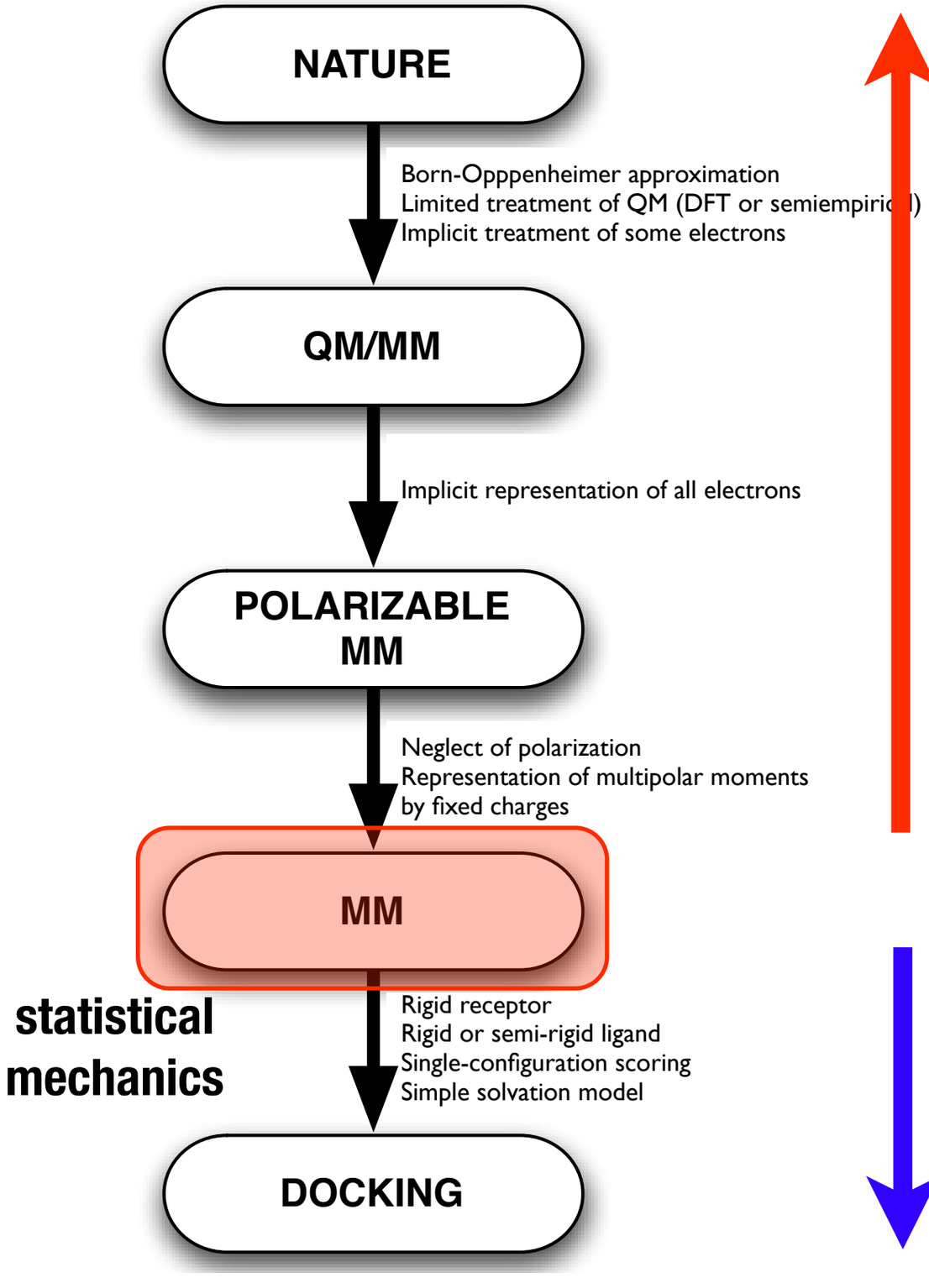
Number of new molecular entities approved by FDA 2005-2009 is half that from previous five years

Number of truly innovative new molecular entities has remained constant at 5-6/year

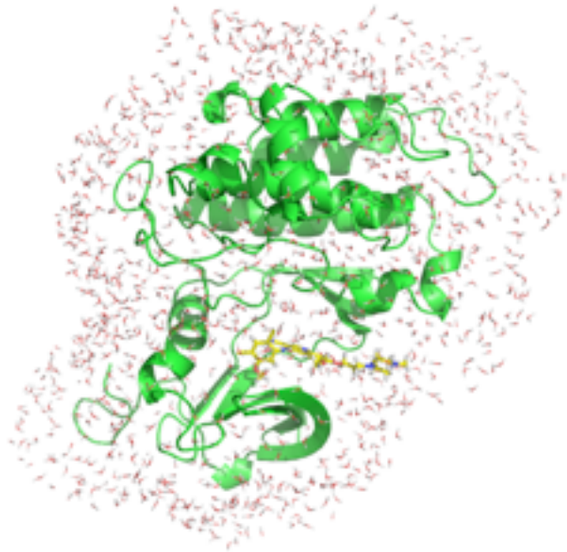


Paul et al. Nat. Rev. Drug Discover. 9:203, 2010.
Chodera et al. Curr. Opin. Struct. Biol., 21:150, 2011.

What details are crucial for useful accuracy in rational design? Our lab is trying to find out!



if insufficiently accurate, add detail in perturbative manner



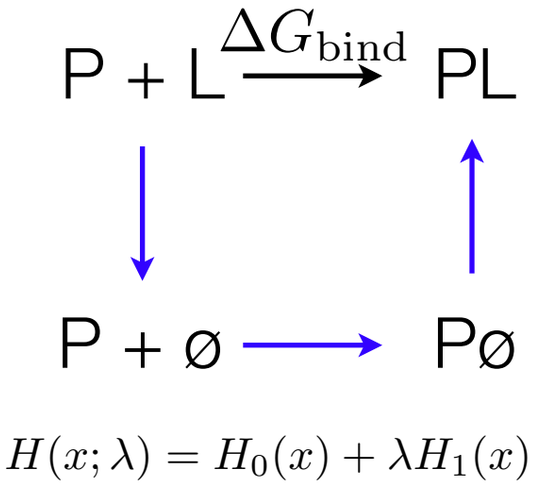
molecular mechanics potential energy function (e.g. AMBER)

$$V(\mathbf{q}) = \sum_{\text{bonds}} K_r(r - r_{eq})^2 + \sum_{\text{angles}} K_\theta(\theta - \theta_{eq})^2$$

$$+ \sum_{\text{dihedrals}} \frac{V_n}{2}[1 + \cos(n\phi - \gamma)] + \sum_{i<j} \left[\frac{A_{ij}}{R_{ij}^{12}} - \frac{B_{ij}}{R_{ij}^{6}} + \frac{q_i q_j}{\epsilon R_{ij}} \right]$$

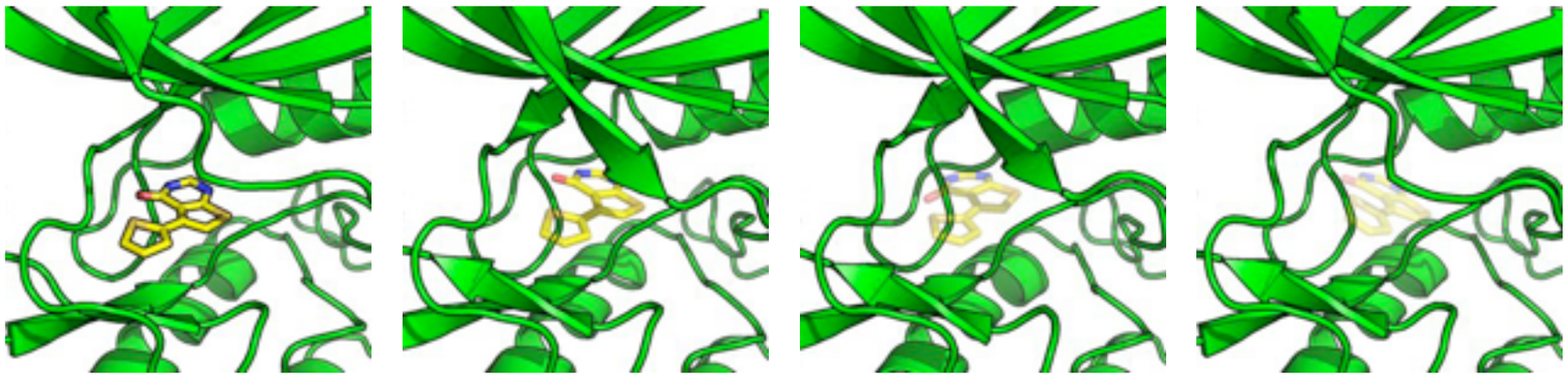
if accurate enough, systematically remove detail

Alchemical free energy calculations provide a rigorous way to efficiently compute binding affinities for a given forcefield



$$H(x;\lambda) = H_0(x) + \lambda H_1(x)$$

multiple simulations of "alchemical intermediates" are run



imposition of restraints | discharging | vdW/steric decoupling | noninteracting

Requires orders of magnitude less effort than simulating direct association process, but includes all enthalpic/entropic contributions to binding free energy.

Not be as fast as OpenEye tools, but will eventually give true binding free energy for a given forcefield (be it bad, or really really bad).

Recent review: Chodera, Mobley, Shirts, Dixon, Branson, Pande. Curr Opin Struct Biol 21:150, 2011.

Learning (to not get depressed) from failure: Fail fast, fail cheap, learn something in the process

computational predictions

experimental confirmation

Predicting binding affinities computationally

$500
1.5 TFLOP
50-500x speedup

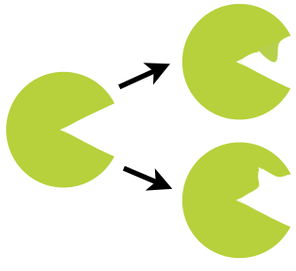
GPU acceleration

$$\pi(x)K(x,y) = \pi(y)K(y,x)$$
$$AH \leftrightharpoons A^- + H^+$$

MCMC framework

enhanced algorithms

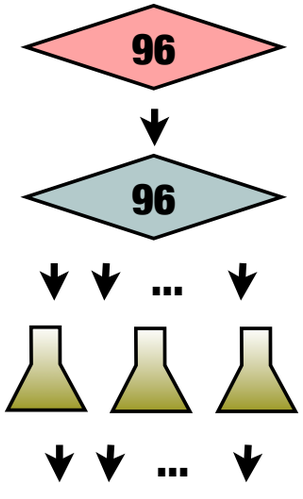
Testing predictions by mutating the target protein



mutate proteins instead of ligands

¢

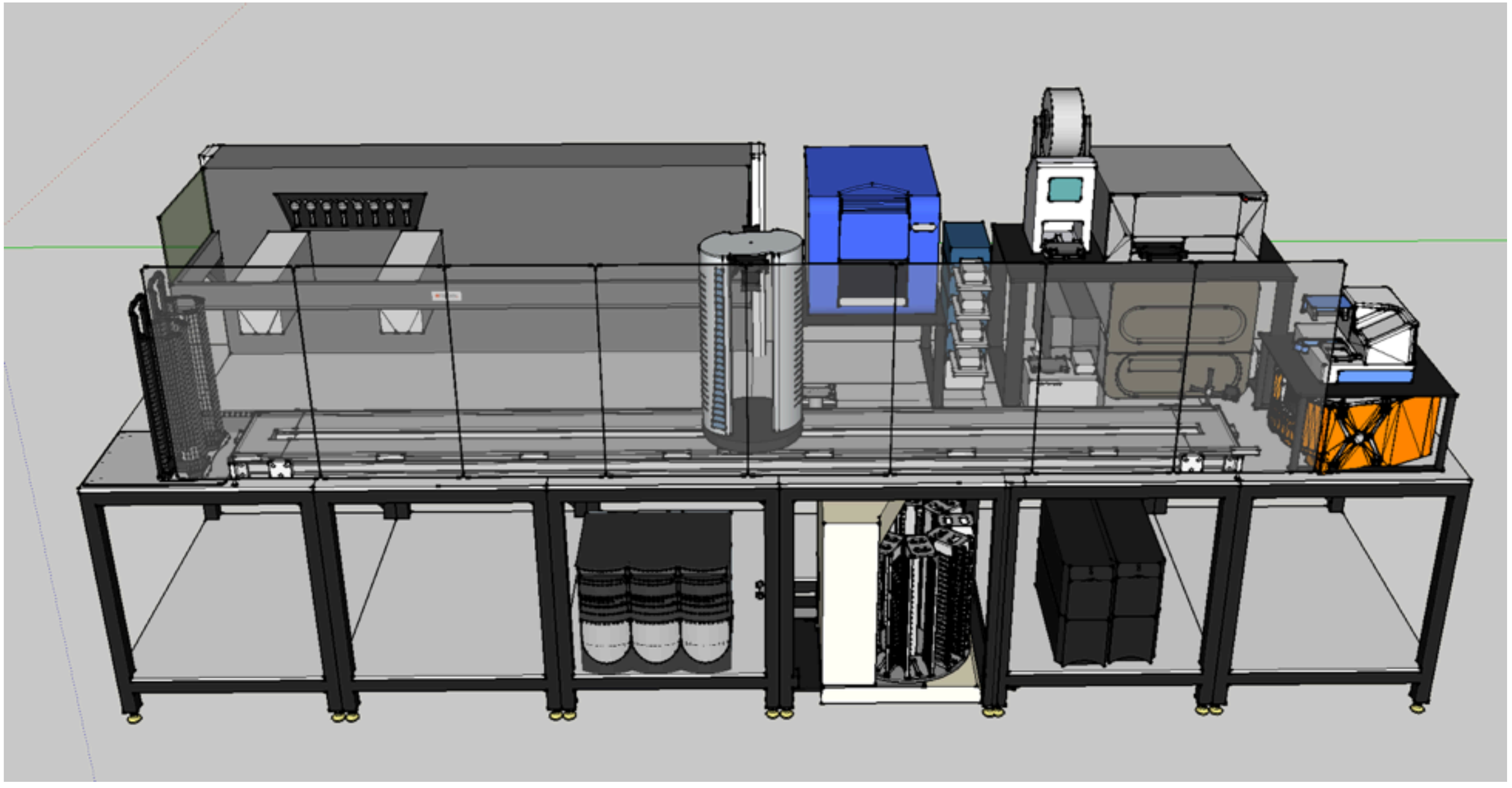
buy inexpensive ligands



high-throughput experiments

The last, best hope for data: RUG-1

(robotic undergraduate no. 1)

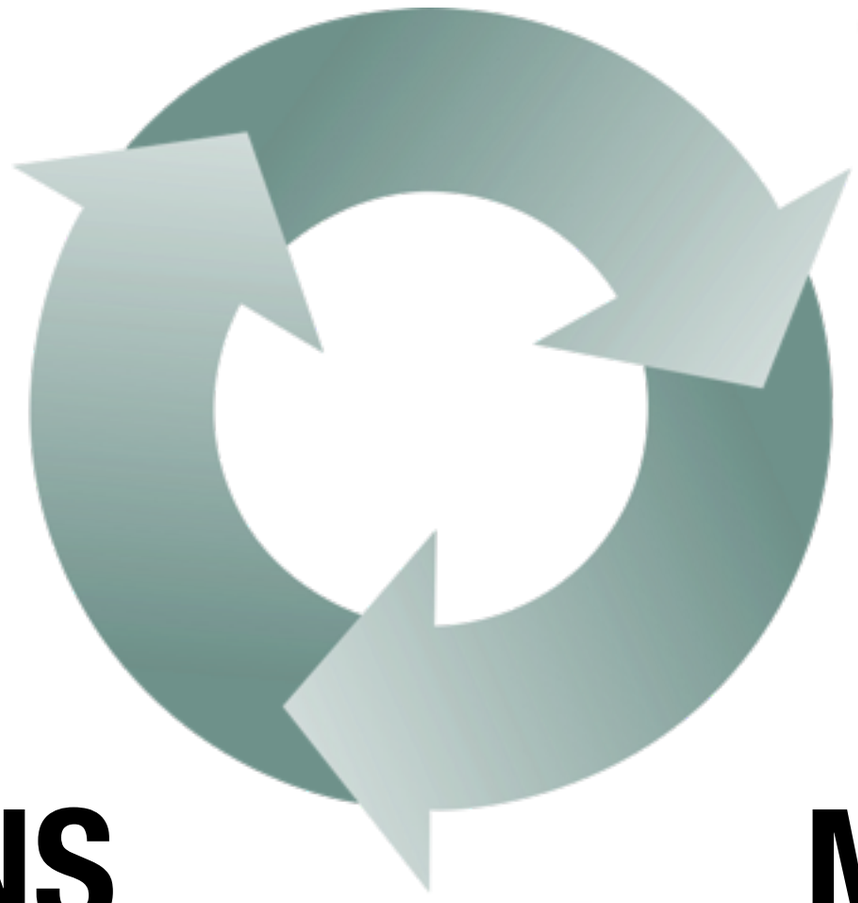


Automated platform for bacterial cloning, mutagenesis, expression, purification, and binding affinity measurement with 24/7 wallet-draining capability

The universal cycle of progress

The universal cycle of progress

DATA

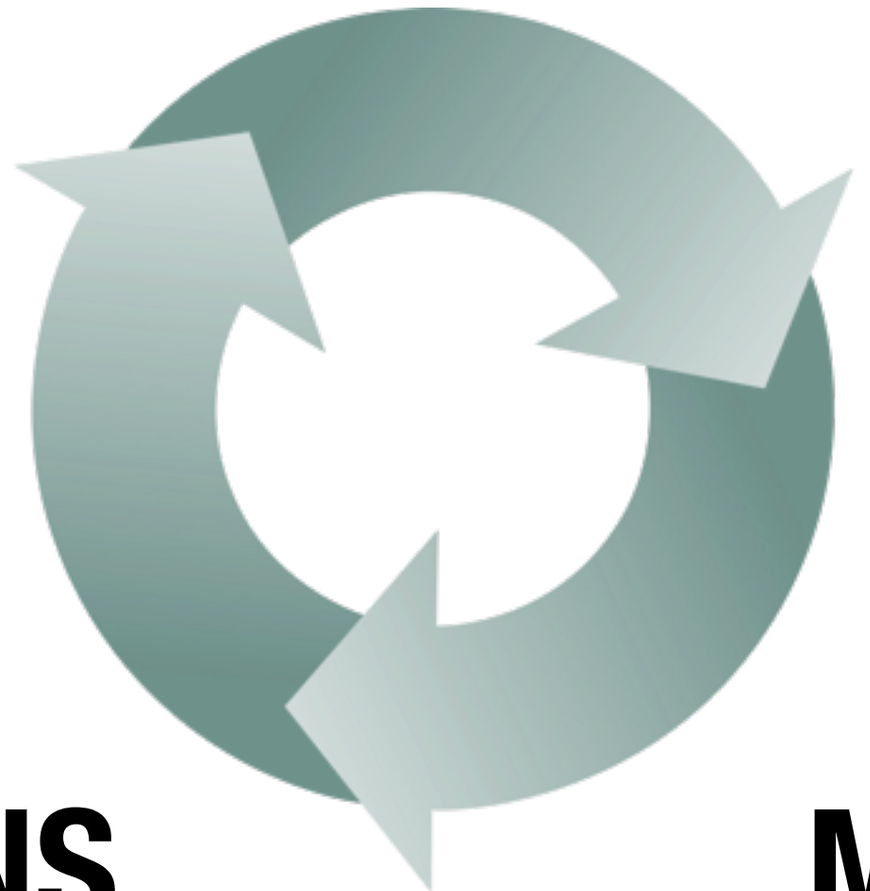


DECISIONS

MODELS

The universal cycle of progress

DATA



DECISIONS **MODELS**

How can we do all this without losing our minds?

Bayesian inference can drive progress

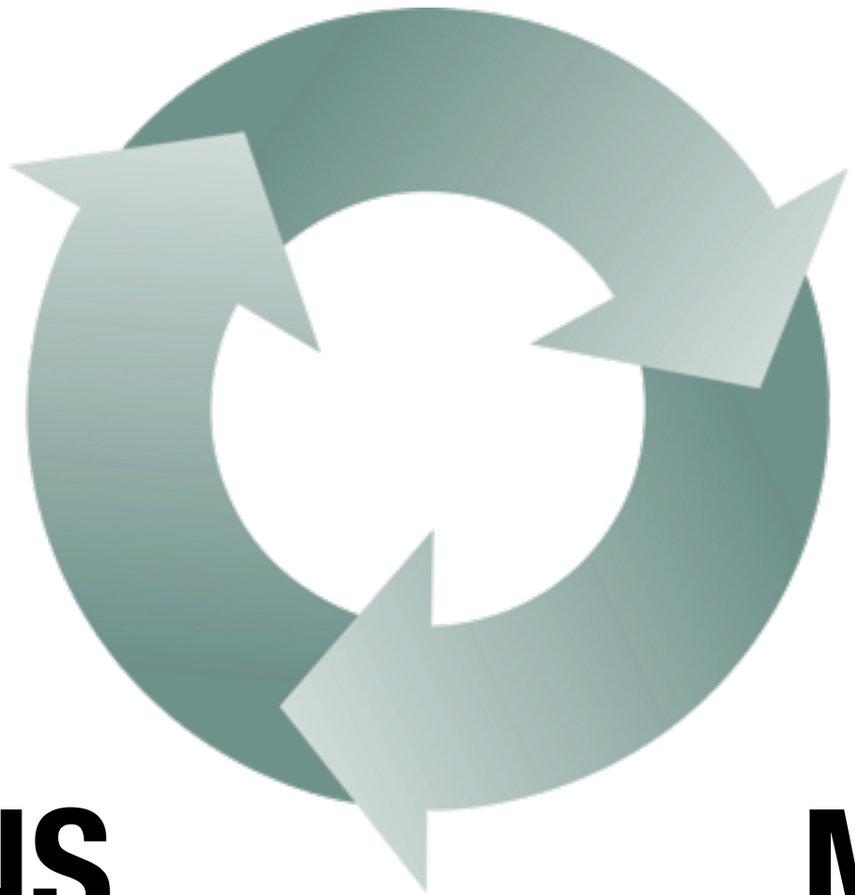
DATA

update knowledge conditioned on data

$$p(\mathcal{I}|\mathcal{D}) \propto p(\mathcal{D}|\mathcal{I})p(\mathcal{I})$$

update model conditioned on data

$$p(\theta|\mathcal{D}) \propto p(\mathcal{D}|\theta)p(\theta)$$



DECISIONS

MODELS

make informed decisions based on model predictions

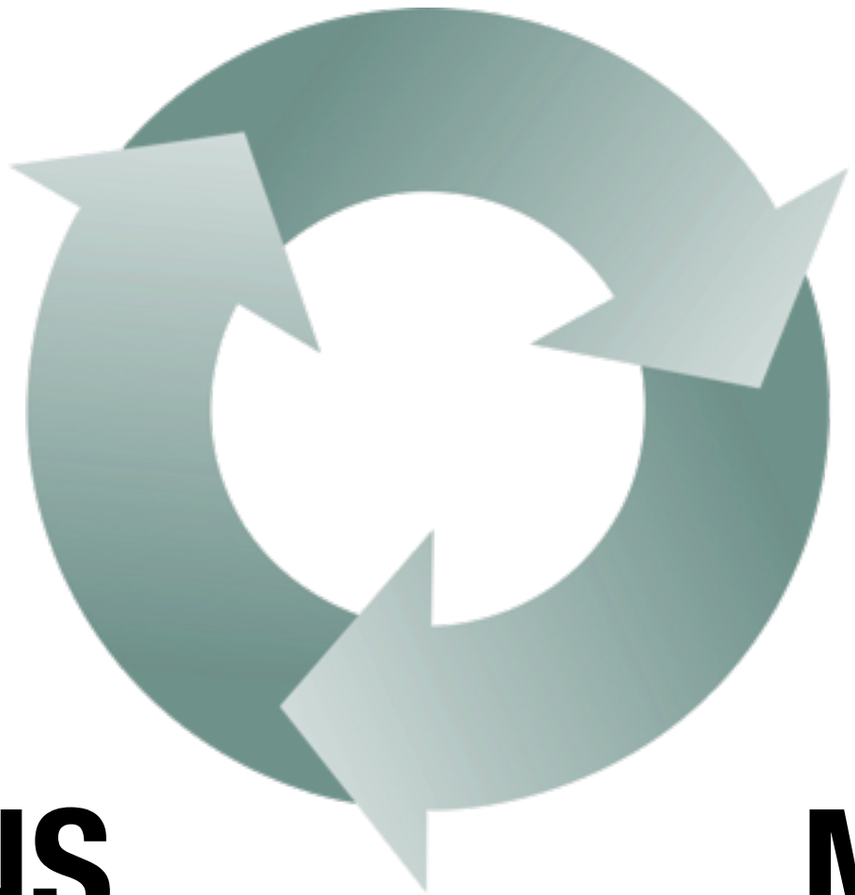
$$\text{argmax}_a \text{E}_a[f(\text{outcome})]$$

Bayesian inference can drive progress

DATA

update knowledge conditioned on data

$$p(\mathcal{I}|\mathcal{D}) \propto p(\mathcal{D}|\mathcal{I})p(\mathcal{I})$$



update model conditioned on data

$$p(\theta|\mathcal{D}) \propto p(\mathcal{D}|\theta)p(\theta)$$

DECISIONS **MODELS**

make informed decisions based on model predictions

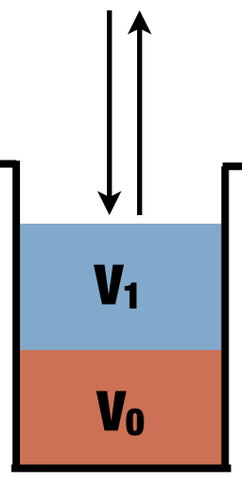
$$\text{argmax}_a \text{E}_a[f(\text{outcome})]$$

What is experimental error and where does it come from?

fluorescence measurement

assay mix

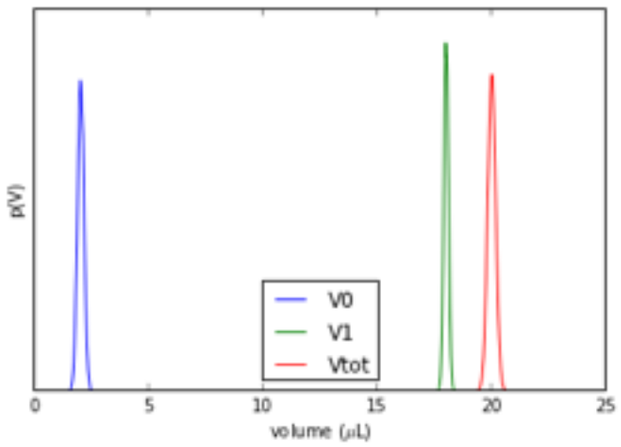
compound



well of plate

$$
\begin{aligned}
V_0 &= 2 \quad \mu L \\
V_1 &= 18 \quad \mu L \\
K_d &= 100 \quad nM \\
C_0 &= 10 \quad mM \\
C_1 &= 1 \quad \mu M
\end{aligned}
$$

finite pipetting precision

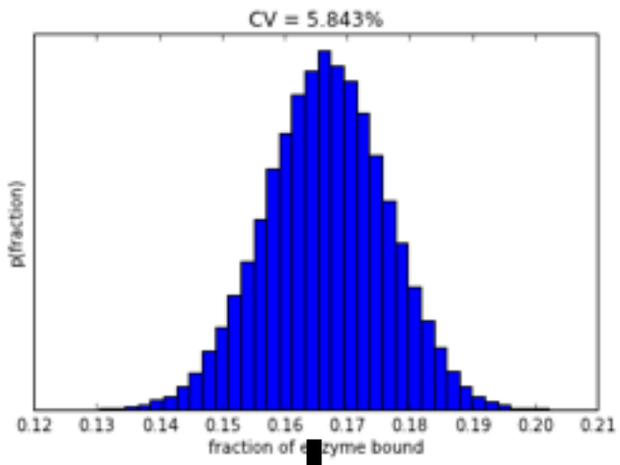


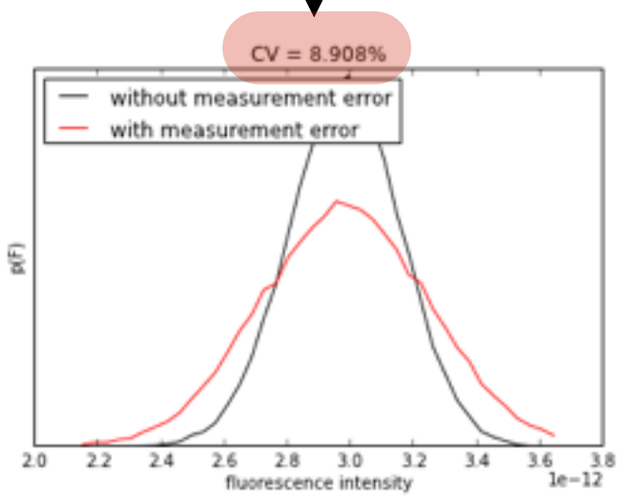
BIOMEK FX® PIPETTING PERFORMANCE SPECIFICATIONS

SPAN-8 SYSTEMS

Transfer Volume	Span-8 Syringe Volume	Tip Types	Accuracy ± %	Precision < %
0.5 µL	250 µL	P20, Fixed 60 mm	5	10
1 µL	250, 500, 1000 µL	P20, P50, Fixed 60 mm	3	7
5 µL	250, 500, 1000 µL	P20, Fixed 60 mm	3	5
10 µL	500 µL	P50, P250	3	5
50 µL	500 µL	P250	3	5
100 µL	500 µL	P250, P1000	3	5

variation in protein:ligand complex concentration

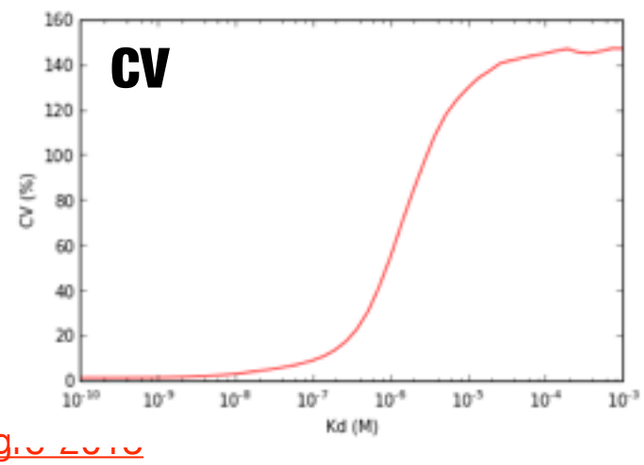




signal broadening due to measurement error

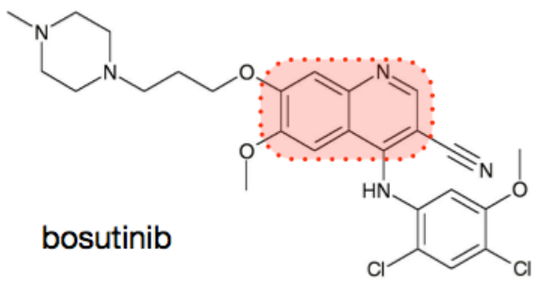
Is the expected CV small enough to be useful?

Modeling an experiment can ensure it will yield useful data or uncover unexpected issues

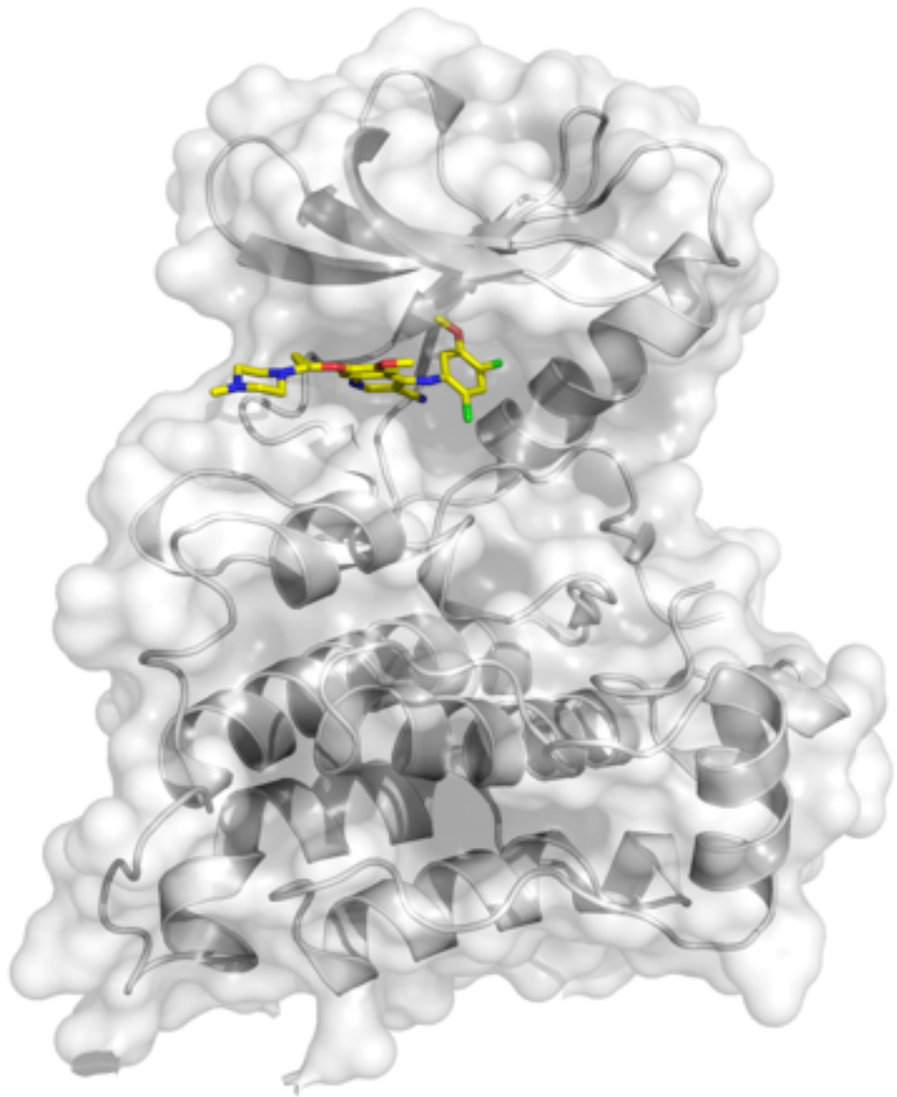


IPython tutorial available at https://github.com/choderalab/cadd-g...

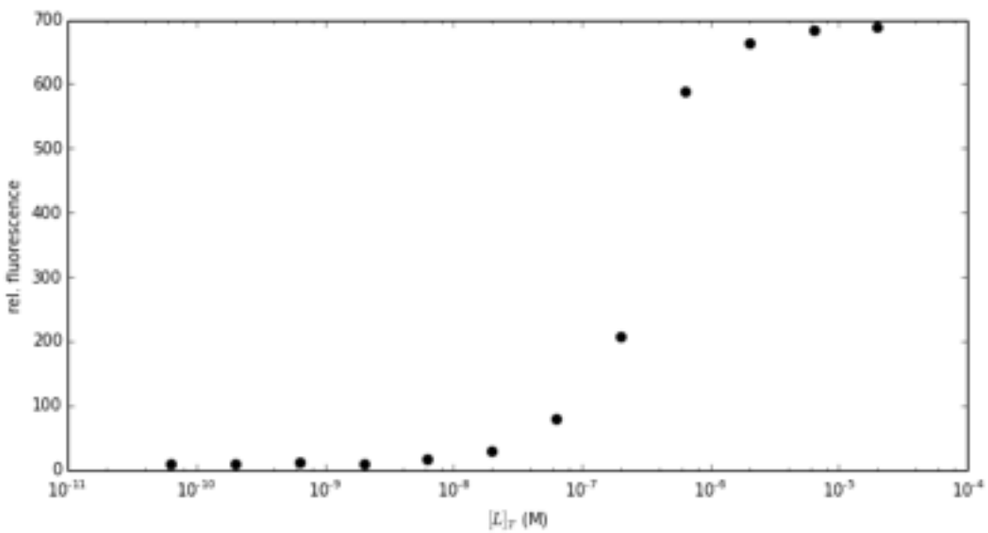
Suppose we have fluorescence data for a dilution series assay of kinase inhibitors



bosutinib

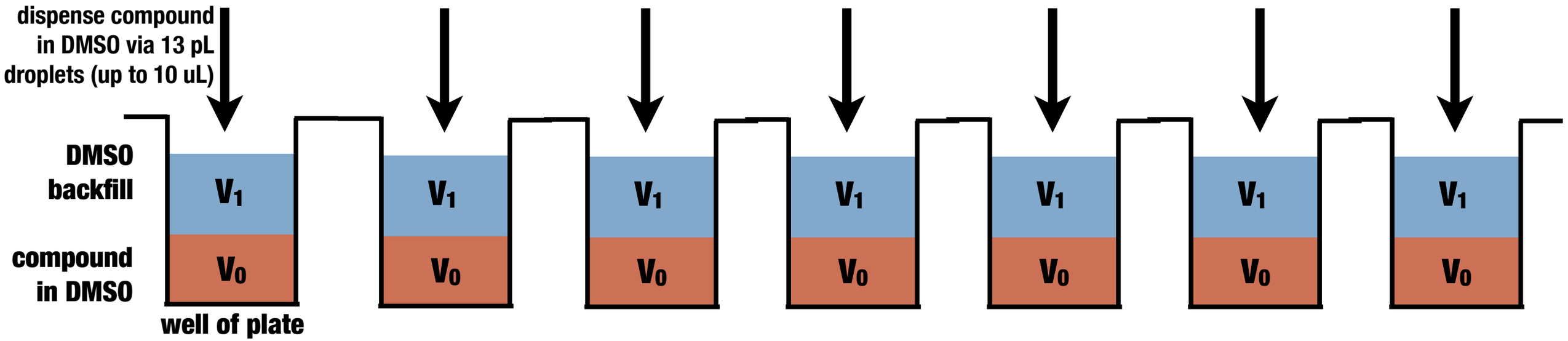


bosutinib bound to Src kinase
RCSB: 4MXO

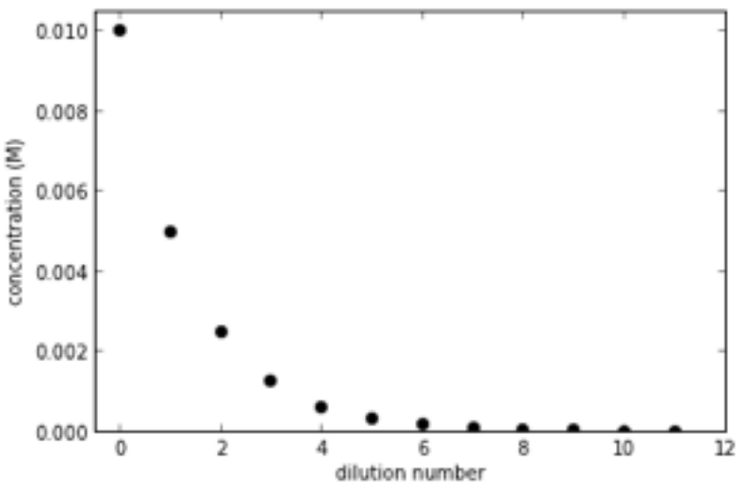


"Structural and Spectroscopic Analysis of the Kinase Inhibitor Bosutinib and an Isomer of Bosutinib Binding to the Abl Tyrosine Kinase Domain", Nicholas M. Levinson, and Steven G. Boxer, PLoS ONE 7, e29828 (2012).

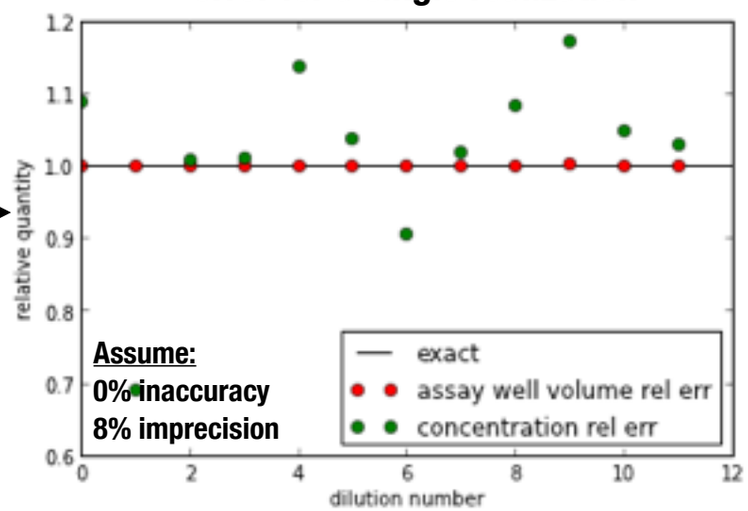
We can use simple Python modeling to propagate the uncertainties in the concentrations of species



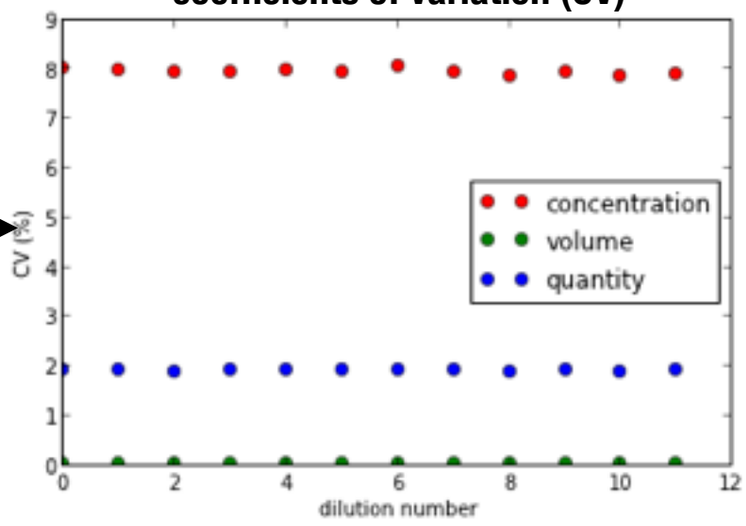
ideal concentrations



error for a single realization



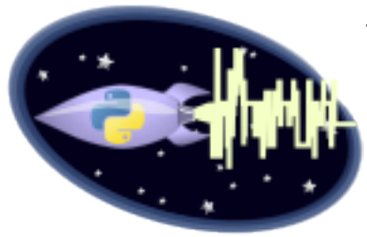
coefficients of variation (CV)



Tecan HP D300

Code at https://github.com/choderalab/cup-xiv

Bayesian inference is easy: pymc is like ‘Bayes in a box’!



https://pypi.python.org/pypi/pymc

binding model

```
import pymc

# Two-component binding.
def two_component_binding(DeltaG, P, L):
    Kd = np.exp(-DeltaG)
    PL = 0.5 * ((P + L + Kd) - np.sqrt((P + L + Kd)**2 - 4*P*L));  # complex concentration (M)
    P = P - PL; # free protein concentration in sample cell after n injections (M)
    L = L - PL; # free ligand concentration in sample cell after n injections (M)
    return [P, L, PL]
```

pymc model

```
# Create a pymc model
def make_model(Pstated, dPstated, Lstated, dLstated, Fobs_i):
    N = len(Lstated)

    # Prior on binding free energies.
    DeltaG = pymc.Uniform('DeltaG', lower=-20, upper=+20, value=0.0) # binding free energy (kT)

    # Priors on true concentrations of protein and ligand.
    Ptrue = pymc.Lognormal('Ptrue', mu=np.log(Pstated**2 / np.sqrt(dPstated**2 + Pstated**2)), tau=np.sqrt(np.log(1.
    Ltrue = pymc.Lognormal('Ltrue', mu=np.log(Lstated**2 / np.sqrt(dLstated**2 + Lstated**2)), tau=np.sqrt(np.log(1.

    # Priors on fluorescence intensities of complexes (later divided by a factor of Pstated for scale).
    F_background = pymc.Gamma('F_background', alpha=0.1, beta=0.1, value=Fobs_i.min()) # background fluorescence
    F_PL = pymc.Gamma('F_PL', alpha=0.01, beta=0.01, value=Fobs_i.max()) # complex fluorescence
    F_L = pymc.Gamma('F_L', alpha=0.01, beta=0.01, value=Fobs_i.max()) # ligand fluorescence

    # Unknown experimental measurement error.
    log_sigma = pymc.Uniform('log_sigma', lower=-3, upper=+3, value=0.0)
    @pymc.deterministic
    def precision(log_sigma=log_sigma): # measurement precision
        return 1.0 / np.exp(log_sigma)

    # Fluorescence model.
    @pymc.deterministic
    def Fmodel(F_background=F_background, F_PL=F_PL, F_P=F_P, F_L=F_L, Ptrue=Ptrue, Ltrue=Ltrue, DeltaG=DeltaG):
        Fmodel_i = np.zeros([N])
        for i in range(N):
            [P, L, PL] = two_component_binding(DeltaG, Ptrue, Ltrue[i])
            Fmodel_i[i] = (F_PL*PL + F_L*L) / Pstated + F_background
        return Fmodel_i

    # Experimental error on fluorescence observations.
    Fobs_i = pymc.Normal('Fobs_i', mu=Fmodel, tau=precision, size=[N], observed=True, value=Fobs_i) # observed data

    # Construct dictionary of model variables.
    pymc_model = { 'Ptrue' : Ptrue, 'Ltrue' : Ltrue,
                   'log_sigma' : log_sigma, 'precision' : precision,
                    'F_PL' : F_PL, 'F_P' : F_P, 'F_L' : F_L, 'F_background' : F_background,
                    'Fmodel_i' : Fmodel, 'Fobs_i' : Fobs_i, 'DeltaG' : DeltaG }
    return pymc_model

# Uncertainties in protein and ligand concentrations.
dPstated = 0.10 * Pstated # protein concentration uncertainty
dLstated = 0.08 * Lstated # ligand concentraiton uncertainty (due to gravimetric preparation and HP D300 dispensing)
```

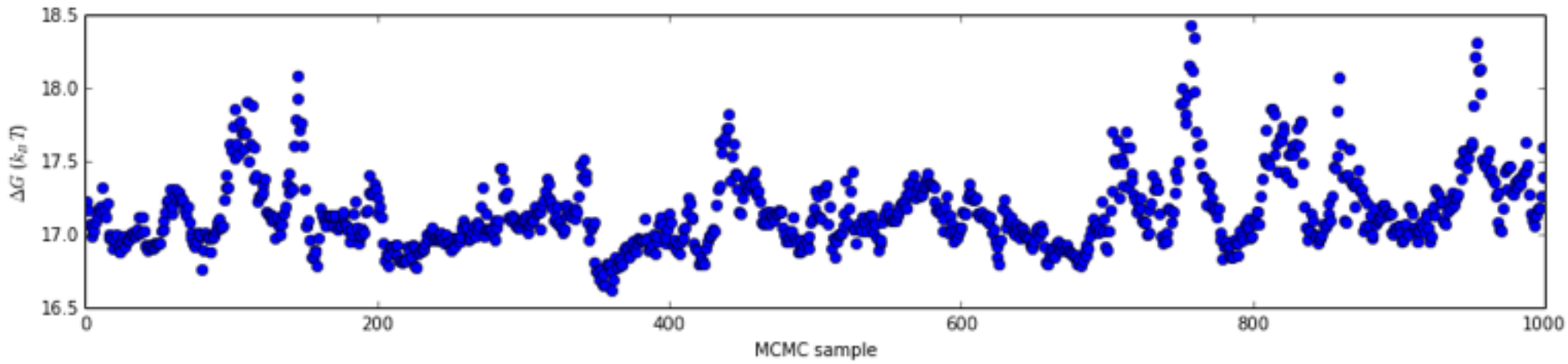
pymc sampling

```
# Build model.
pymc_model = pymc.Model(make_model(Pstated, dPstated, Lstated, dLstated, F_i))

# Sample with MCMC
mcmc = pymc.MCMC(pymc_model, db='ram', name='Sampler', verbose=True)
mcmc.sample(iter=100000, burn=50000, thin=50, progress_bar=False)
```

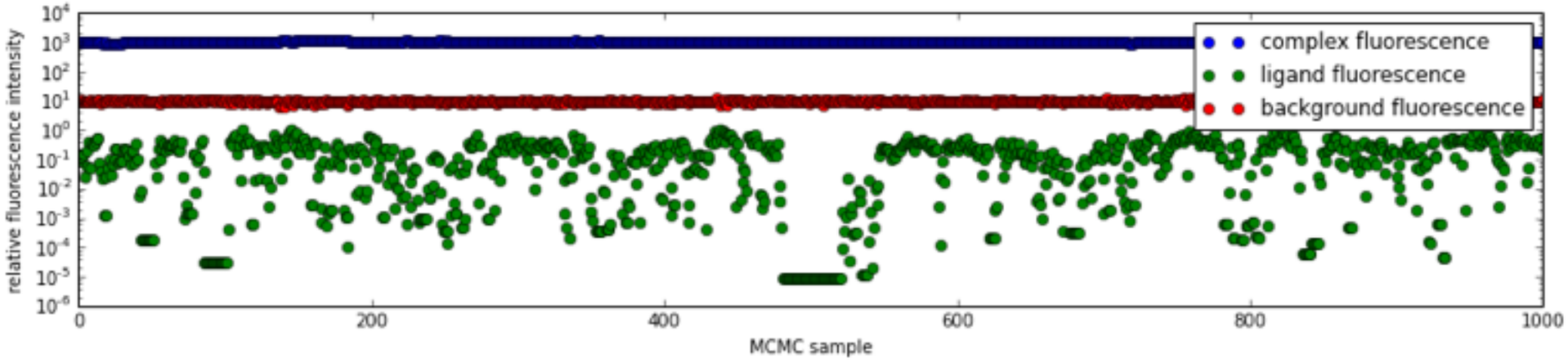
Code at https://github.com/choderalab/cup-xiv

Suppose we have fluorescence data for a dilution series assay of kinase inhibitors

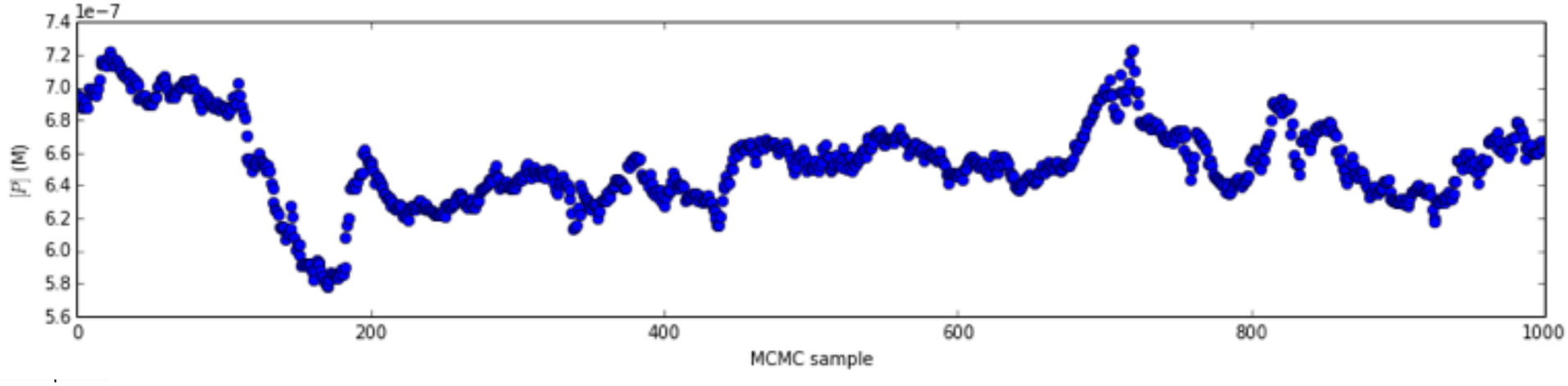
MCMC sampling of free energies



MCMC sampling of intrinsic fluorescence



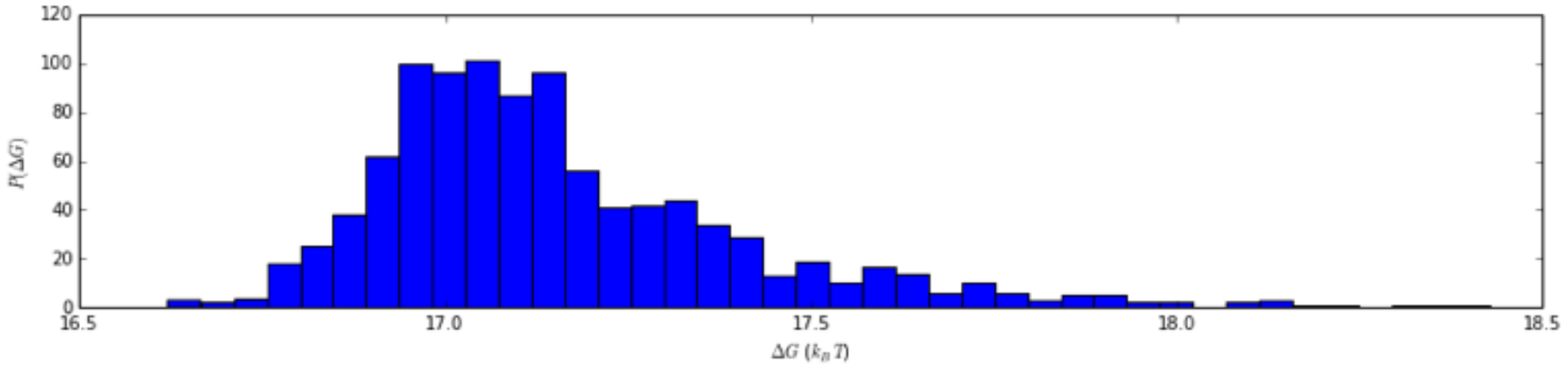
MCMC sampling of true concentrations



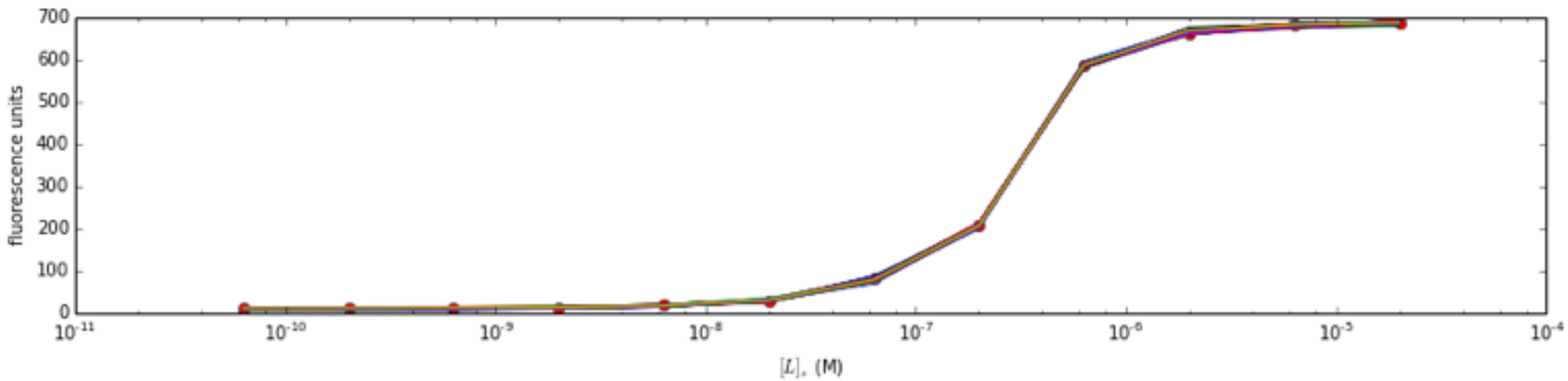
Code at https://github.com/choderala

Suppose we have fluorescence data for a dilution series assay of kinase inhibitors

posterior distribution of binding free energies



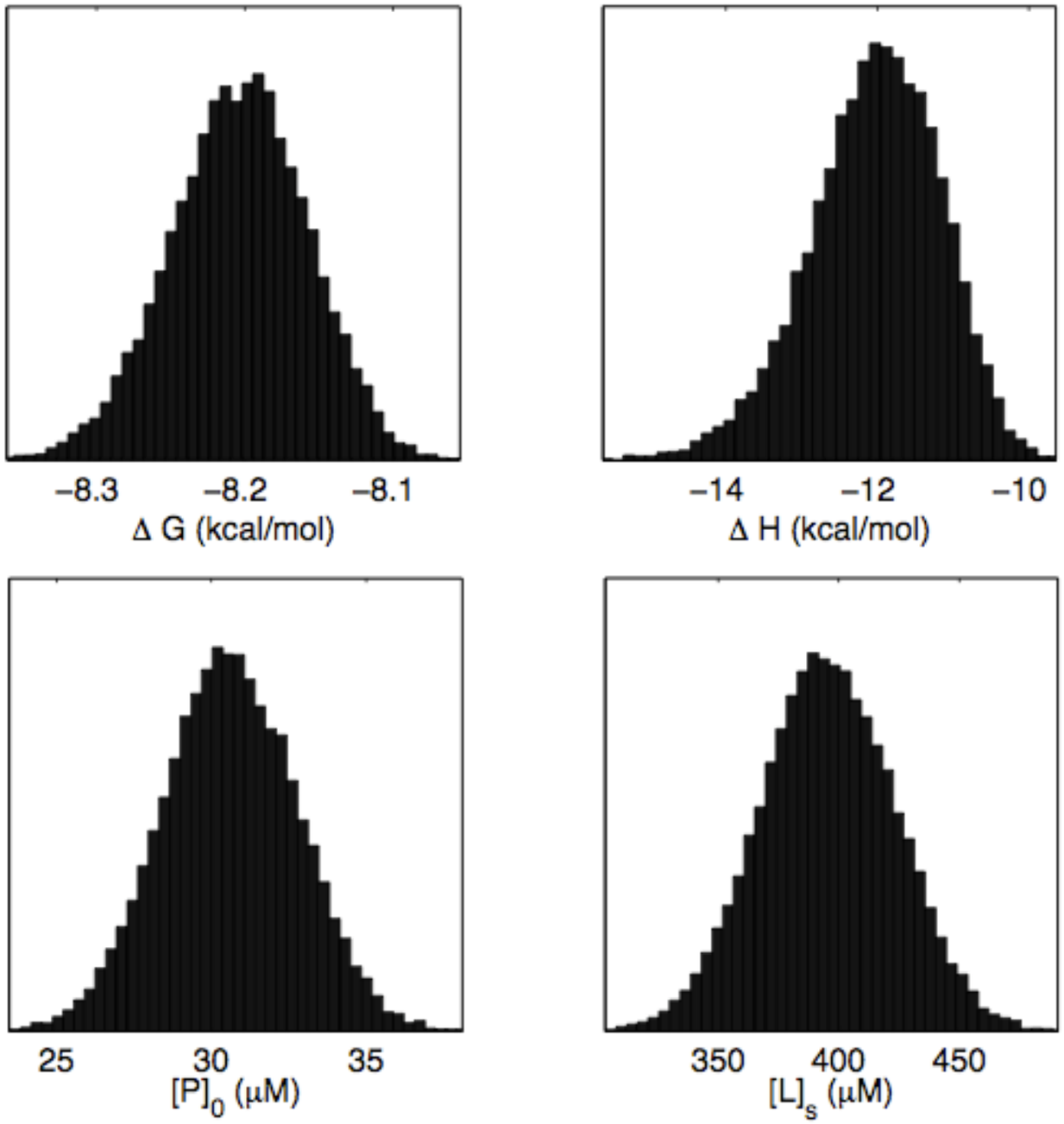
family of models fit to experimental fluorescence data

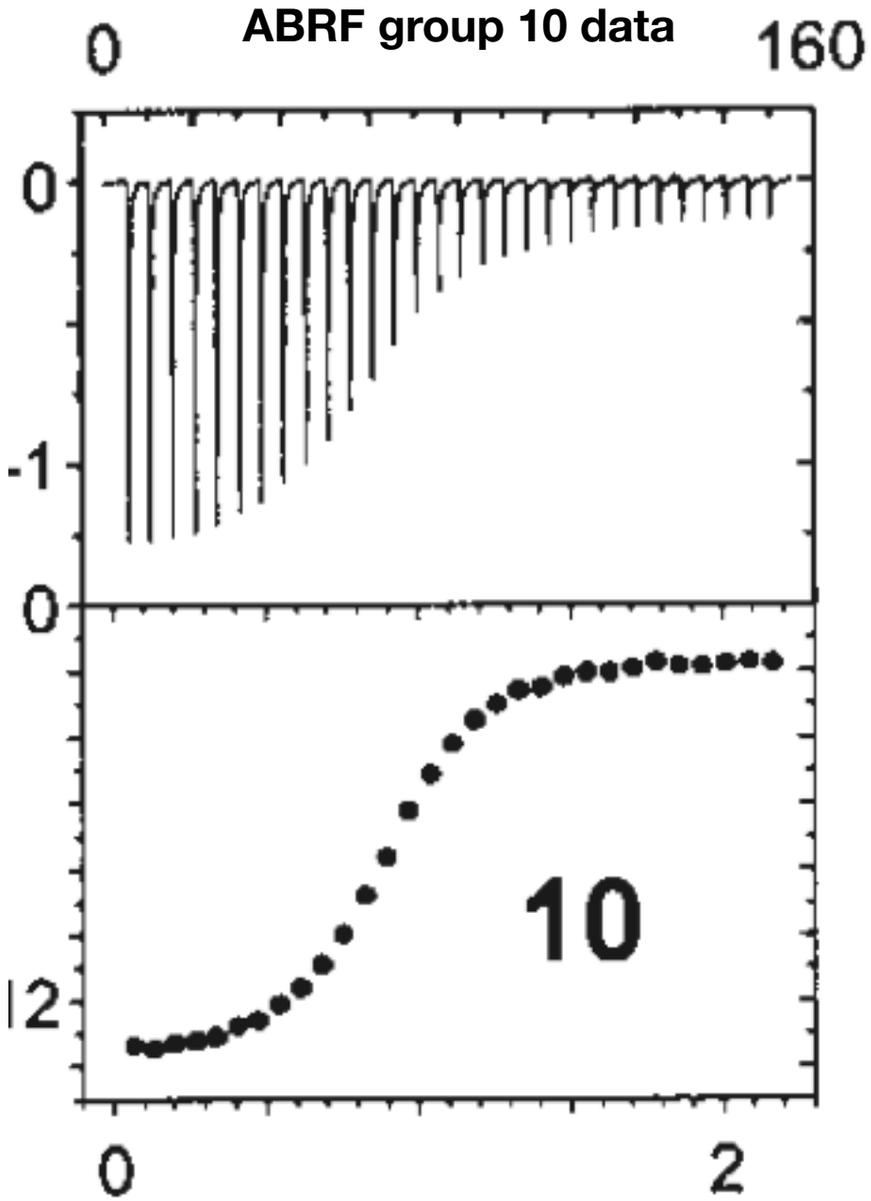


Code at https://github.com/choderalab/cup-xiv

Analysis of ITC experiments: The Bayesian way

Marginal posterior probability distributions describe the uncertainty in each thermodynamic parameter inferred from the data.

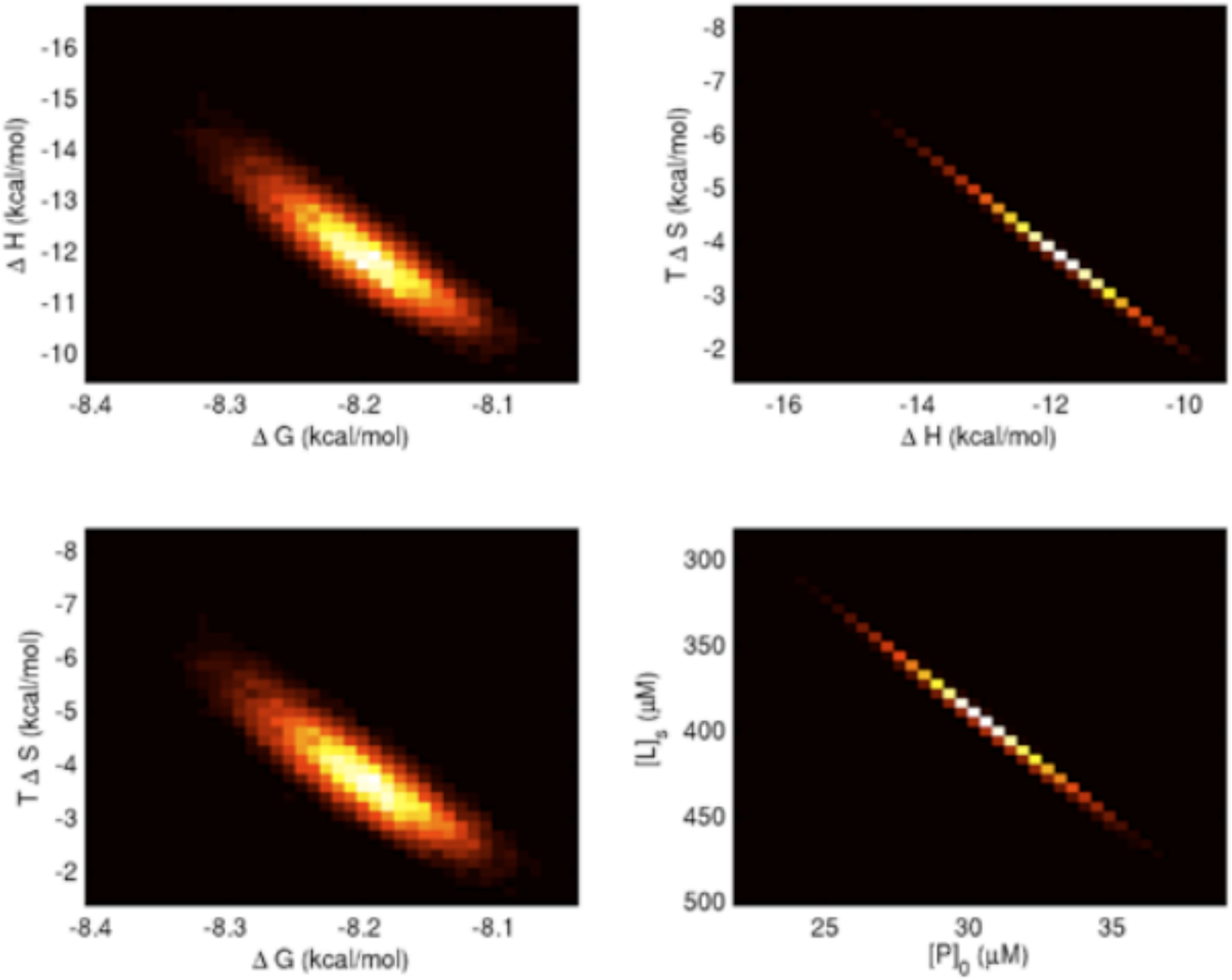




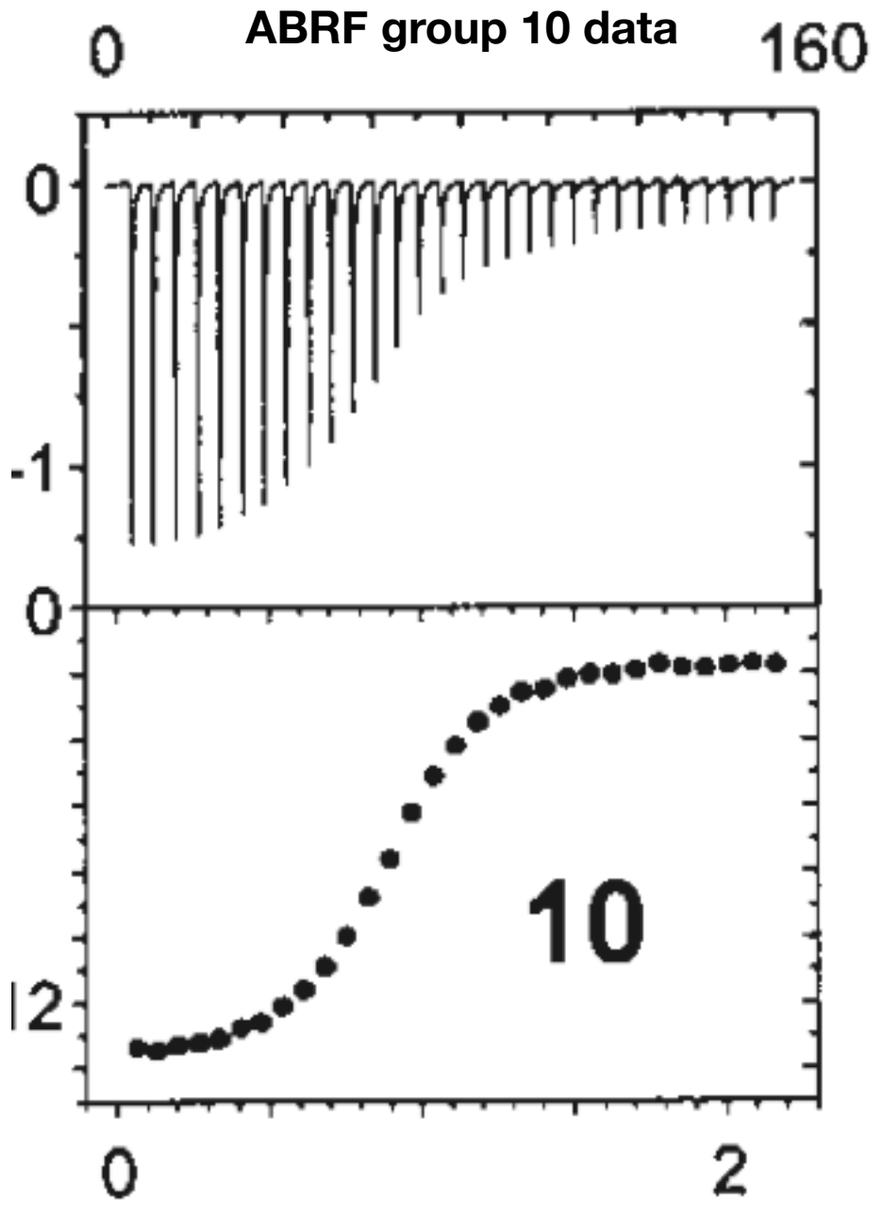
Myszka et al. J. Biomol. Tech. 14:247, 2003.

Analysis of ITC experiments: The Bayesian way

Joint distribution functions describe the correlated uncertainty between any set of parameters.



Correlation in error in ΔH and TΔS explains much of apparent entropy-enthalpy compensation behavior.



Myszka et al. J. Biomol. Tech. 14:247, 2003.

Bayesian data analysis has numerous advantages

Provides true posterior joint distribution of all thermodynamic parameters

Asymmetric confidence intervals and non-normal marginal distributions

Easy to "plug in" new binding models.

Make joint inferences from multiple experiments

Instrument parameters can be conditioned on calibration data (e.g. standard titrations)

Expected information content of new experiments can be estimated for protocol design

Experimental design:

> **"Will experiment X give me enough information to make it worthwhile?"**
> **"What is the best experimental design to reduce the uncertainty in Z?"**
> **"Do I have to run a baseline for sample X?"**

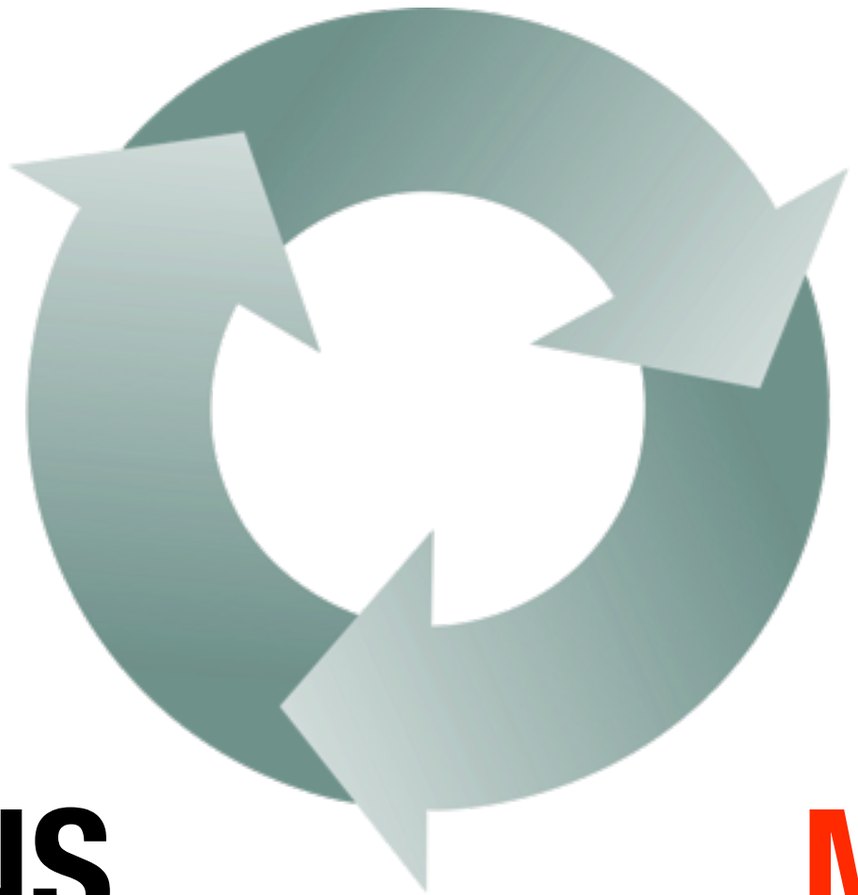
Code will be available at http://github.com/choderalab/bayesian-itc

The universal cycle of progress
Bayesian inference can drive progress

DATA

update knowledge conditioned on data

$$p(\mathcal{I}|\mathcal{D}) \propto p(\mathcal{D}|\mathcal{I})p(\mathcal{I})$$



update model conditioned on data

$$p(\theta|\mathcal{D}) \propto p(\mathcal{D}|\theta)p(\theta)$$

DECISIONS

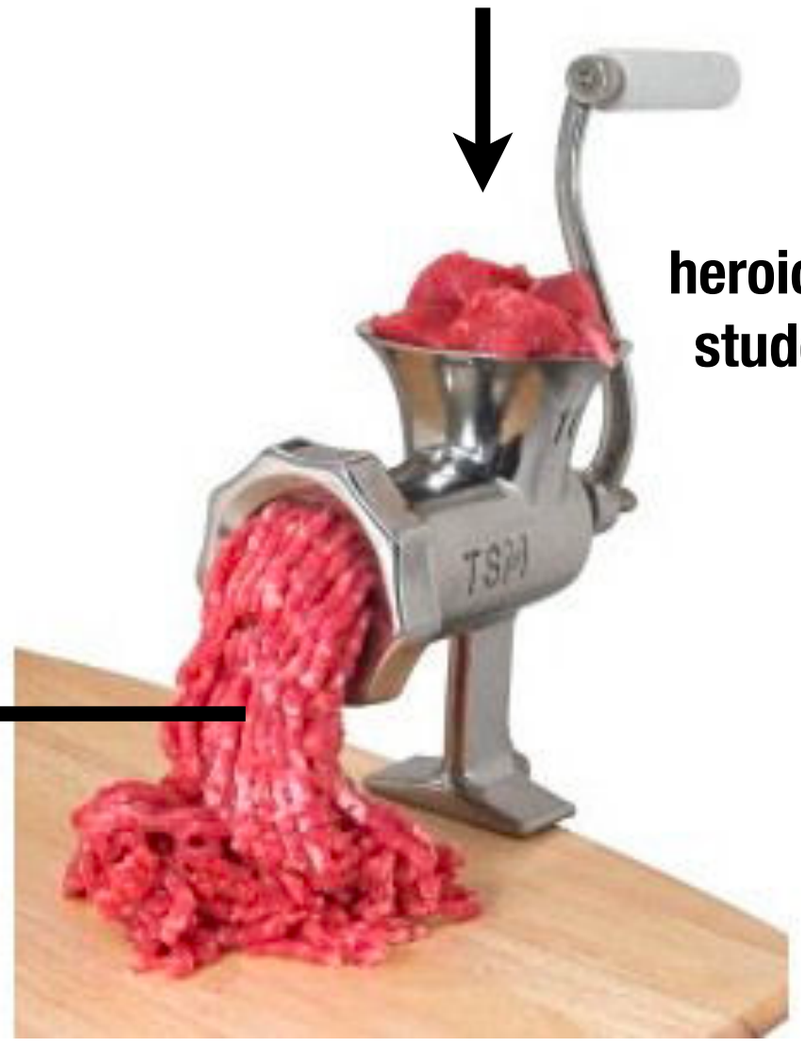
MODELS

make informed decisions based on model predictions

$$\text{argmax}_a \text{E}_a[f(\text{outcome})]$$

How are forcefields made?

experimental data
quantum chemistry
keen chemical intuition



heroic effort by graduate students and postdocs

a parameter set we desperately hope someone actually uses

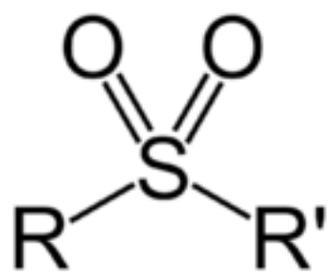
Poorly represented functional groups are problematic

Hydration free energy errors by functional group

functional group	number	*t*-value	significance	mean error
acid	73	−7.43	4e-13	−0.34
alcohol	38	3.62	0.0003	1.29
aldehyde	20	−3.04	0.003	−0.07
alkanes	28	−1.69	0.09	0.31
alkene	35	2.34	0.02	1.07
alkyl bromide	17	3.31	0.001	1.50
alkyl chloride	31	2.31	0.02	1.09
alkyl iodide	9	0.59	0.6	0.86
alkyne	6	−0.38	0.7	0.49
amine	44	−0.65	0.5	0.55
aromatic compound	170	−1.05	0.3	0.55
aryl chloride	20	1.65	0.1	1.04
carbonitrile	12	3.22	0.001	1.63
cyclic hydrocarbon	8	−1.18	0.2	0.21
ester	8	−1.69	0.09	0.02
ether	42	2.18	0.03	1.01
halogen derivative	22	0.32	0.8	0.73
heterocyclic compound	48	2.38	0.02	1.02
hypervalent S	5	−4.55	7e-06	−1.50
ketone	25	−2.77	0.006	0.05
nitro compound	17	1.86	0.06	1.13
other	29	−0.48	0.6	0.55
phenol or hydroxyhetarene	33	2.72	0.007	1.16
thiol	5	0.51	0.6	0.89

Mobley DL, Bayly CI, Cooper MD, Shirts MR, Dill KA. JCTC 5:350, 2009.

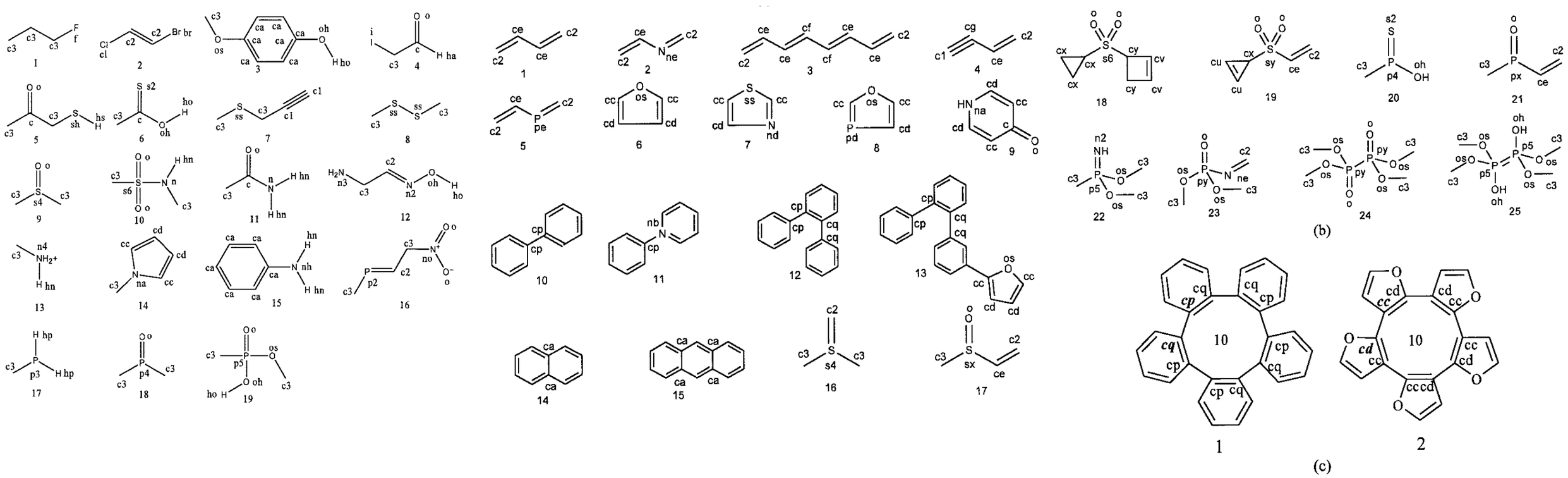
Poorly represented functionalities in parameterization set lead to pathologies in use.



sulfonyl moiety

As drug discovery explores new parts of chemical space, how can forcefields keep up?

GAFF was parameterized for a limited diversity of compounds (and hence atom types):



Extension of GAFF by others “nontrivial” (effectively impossible for anyone but Junmei)

Wang J, Wolf RM, Caldwell JW, Kollman PA, and Case DA. J Comput Chem 25:1157, 2004.

Our approach to parameterization has evolved over time, but it's still not completely automated by any measure

year	forcefield	parameter fitting	atom types
1990s	AMBER parm96	lots of "hand tweaking"	hand-picked
early 2000s	GAFF	genetic algorithm	hand-picked
mid 2000s	TIP4P-Ew	least-squares optimization	hand-picked

(an AMBER-centric view, but meant to be illustrative)

Torsion barrier for peptide bond from parm96.dat

```
X -C -N -X    4   10.00        180.0                2.          AA|check Wendy?&NMA
```

How can we move to entirely automated schemes that are easy to grow and refine?

What would we want out of a forcefield parameterization scheme (while we're dreaming)?

Everything is automatic; don't need to tweak things by hand.

There is only one Christopher Bayly, and he's pretty busy, so we shouldn't need to consult him too frequently for deep feats of chemical insight.

Automatically chooses optimal functional forms.

If we find ourselves in uncharted territory in chemical space, can add more data. (Maybe it could even tell us which new data would be useful!)

I am lazy and don't want to learn new algorithms.

Would give us an idea of how reliable it new predictions are expected to be.

Is there really a procedure that could fit these criteria?

Bayesian inference can help!

Bayes rule provides a probability measure over unknown parameters given data:

$$p(\theta|\mathcal{D}) \propto p(\mathcal{D}|\theta)p(\theta)$$

$\mathcal{D}$ data

θ forcefield parameters

$p(\theta|\mathcal{D})$ posterior

$p(\mathcal{D}|\theta)$ data model

$p(\theta)$ prior on forcefield parameters

Bayesian parameterization can exploit the one tool we already know how to use well, Markov chain Monte Carlo

---

1. experimental/QM data
2. uncertainties/errors
3. spectroscopic model

$$p(\theta|\mathcal{D}) \propto p(\mathcal{D}|\theta)p(\theta)$$

MCMC

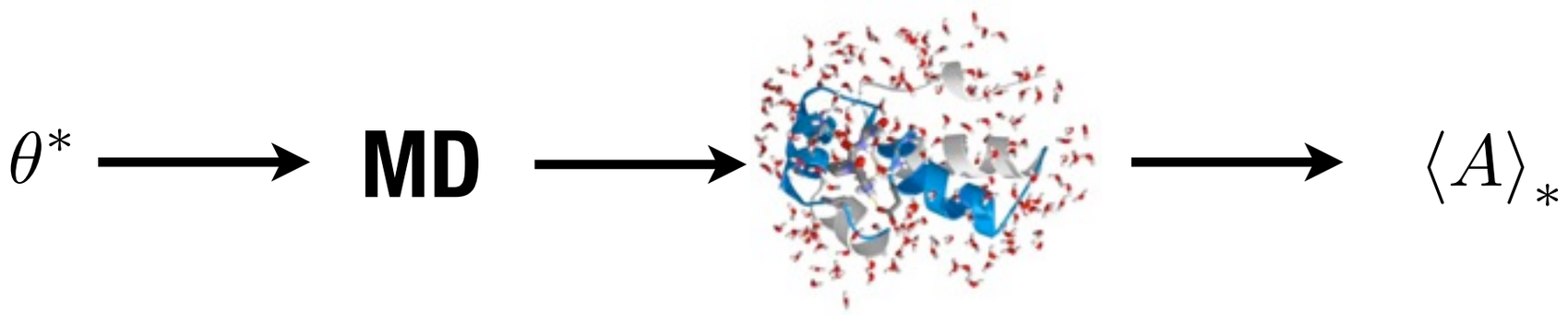
multiple parameter sets consistent with data

$$U_{\text{eff}}(\theta) \equiv -kT \log p(\theta|\mathcal{D})$$

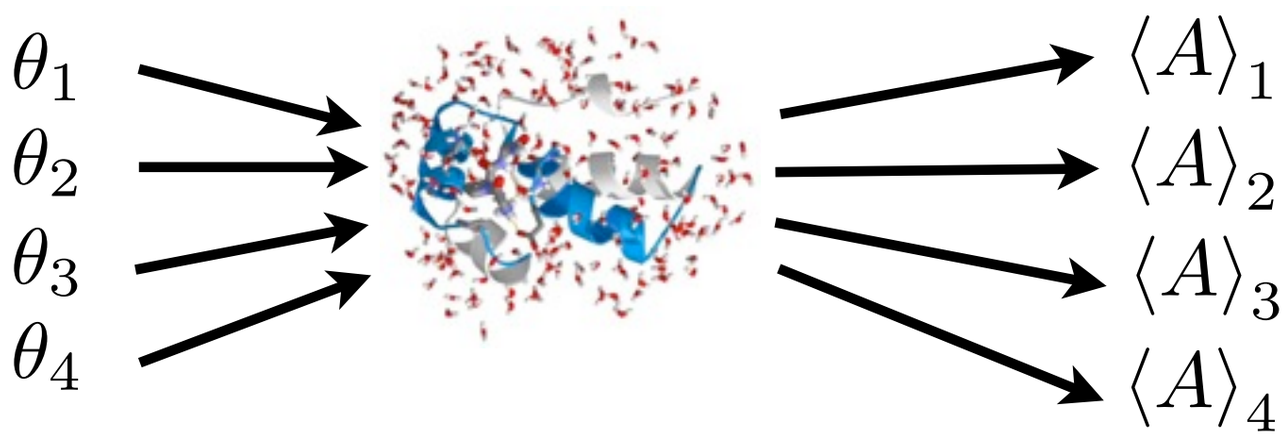
effective energy function

Evaluation of computed properties with multiple parameter sets gives an estimate of systematic error in forcefield

Compute properties with one representative parameter set from collection



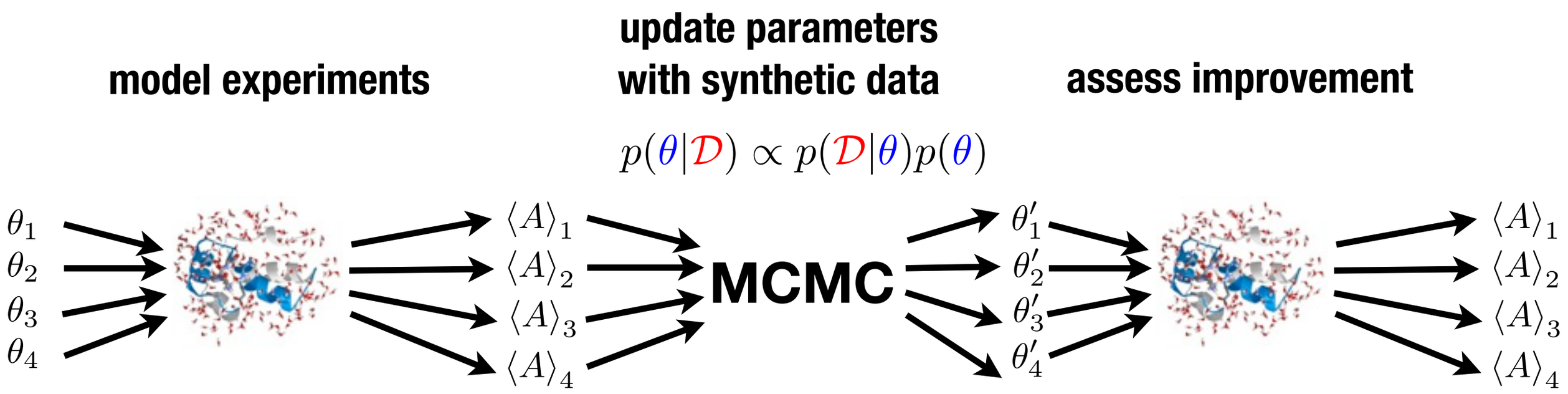
Estimate computed properties for other parameter sets from collection by reweighting (e.g. MBAR)



Can estimate both statistical and systematic components of computed property!

Bayesian experimental design can tell us which kinds of data would best help reduce systematic error

---



Modeling the outcome of new experiments can suggest which will be most informative in reducing systematic error

Bayesian model selection can be a powerful tool in forcefield development and parameterization

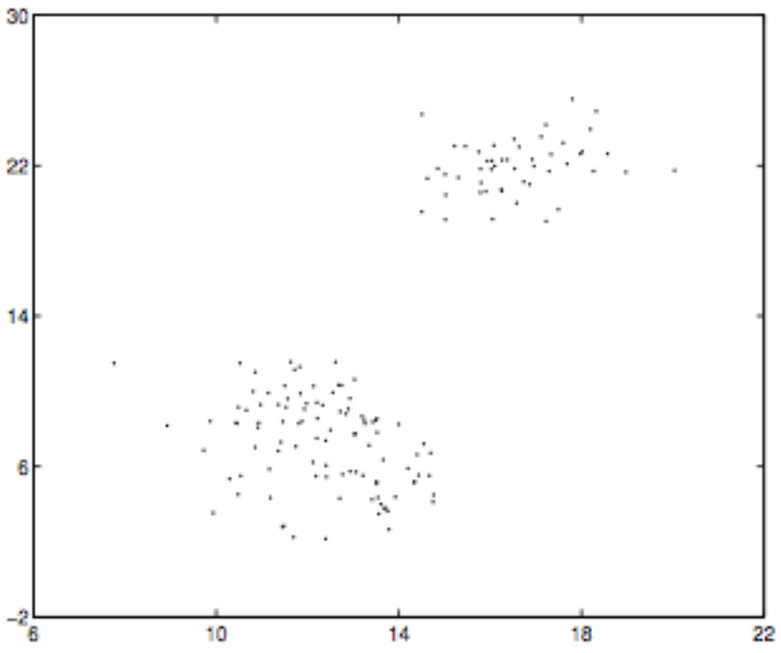
---

Classic Gaussian mixture model case:

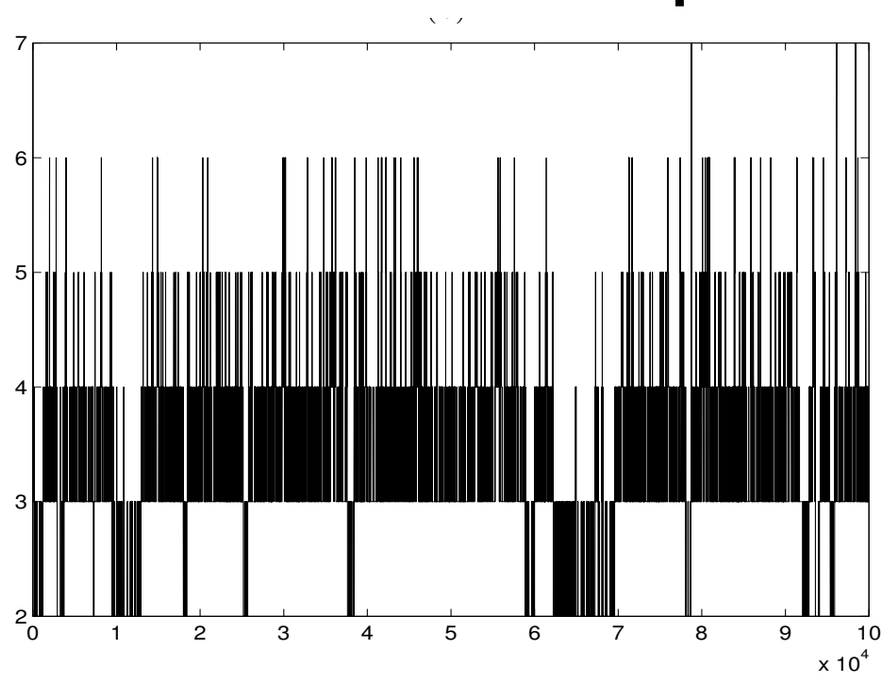
- How many components do the data support?

- What are their parameters?

data



number of Gaussian components



Bayesian inference penalizes complexity

Bayesian model selection can be a powerful tool in forcefield development and parameterization

---

Reversible-jump Monte Carlo (RJMC) allows jumps among models; models best supported by data are preferentially sampled. (Calculation is analogous to grand canonical MC.)
RJMC moves simply have to satisfy detailed balance in model space (e.g. via Metropolis-Hastings):

$$P_{\text{accept}} = \min\left\{1, \frac{P(\text{new})}{P(\text{old})} \cdot \frac{P(\text{old}|\text{new})}{P(\text{new}|\text{old})}\right\}$$

RJMC can automate selection of **most appropriate functional form** and/or **mixing rules** for vdW:

Lennard-Jones	Lorentz-Berthelot
exponential-6	Waldman-Hagler
Halgren buffer 14-7	Halgren HHG

RJMC can automate selection of **atom types**: only as many types as the data supports!

- splitting moves that split off a subtype (e.g. sp3 carbon split from generic carbon type)
- subtype deletion moves that remove subtypes

What would we want out of a forcefield parameterization scheme?

- Everything is automatic; don't need to tweak things by hand.
  Data goes in, parameters come out!
- Never need to choose atom types or perform immense feats of chemical intuition.
  Bayesian model selection with RJMC!
- If we're uncertain about which functional forms are best, automatically chooses.
  Bayesian model selection with RJMC!
- If we find ourselves in uncharted territory in chemical space, throw in more data. (Maybe it could give us hints about which data would be useful to obtain?)
  Bayesian updating and experimental design!
- Would be nice if parameterization procedure didn't require me to learn crazy new mathematics or algorithms (mainly because I'm lazy)
  Isomorphic with statistical mechanics!
- Would give us an idea of how reliable it new predictions are expected to be.
  Multiple parameter sets from Bayesian posterior

The future of forcefield parameterization?

experimental data
quantum chemistry
uncertainties

ensemble of
parameter sets

pull the trigger and go!

The universal cycle of progress
Bayesian inference can drive progress

DATA

update knowledge conditioned on data

$$p(\mathcal{I}|\mathcal{D}) \propto p(\mathcal{D}|\mathcal{I})p(\mathcal{I})$$

update model conditioned on data

$$p(\theta|\mathcal{D}) \propto p(\mathcal{D}|\theta)p(\theta)$$

DECISIONS

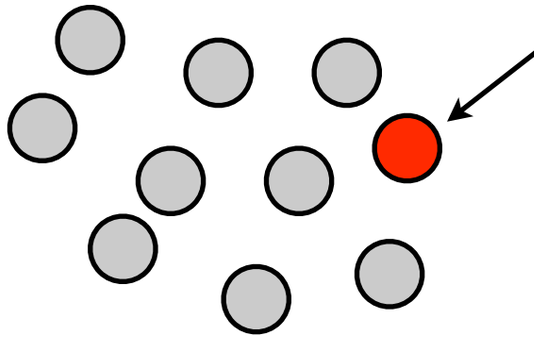
MODELS

make informed decisions based on model predictions

$$\text{argmax}_a \text{E}_a[f(\text{outcome})]$$

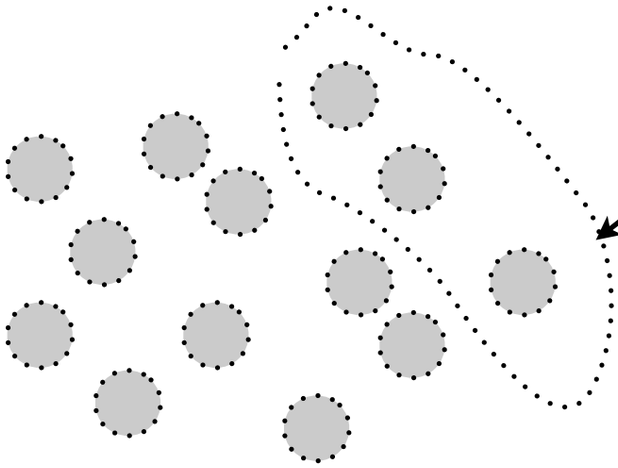
“Pick a winner”

Suppose we make predictions for 20 compounds the chemists can make. They don’t want to make all 20.



best one so far

some compounds we’ve made



how many the chemists want to make

a bunch of compounds we could make

What should they make?

Code at https://github.com/choderalab/cup-xiv

Let's pick a simple illustrative model

Simple model for affinity/potency/whatever:

$$\Delta G = \mathbf{x}^{\mathrm{T}} \mathbf{A} \mathbf{x}$$

feature/fingerprint vector $\mathbf{x} = [x_1 \; x_2 \; x_3]^{\mathrm{T}}$

model parameters $\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$

true model $\mathbf{A}^*$

error in measured affinities $\Delta G_{obs} = \Delta G + \xi$

measurement error is Gaussian with zero mean $\xi \in \mathcal{N}(0, \sigma^2)$

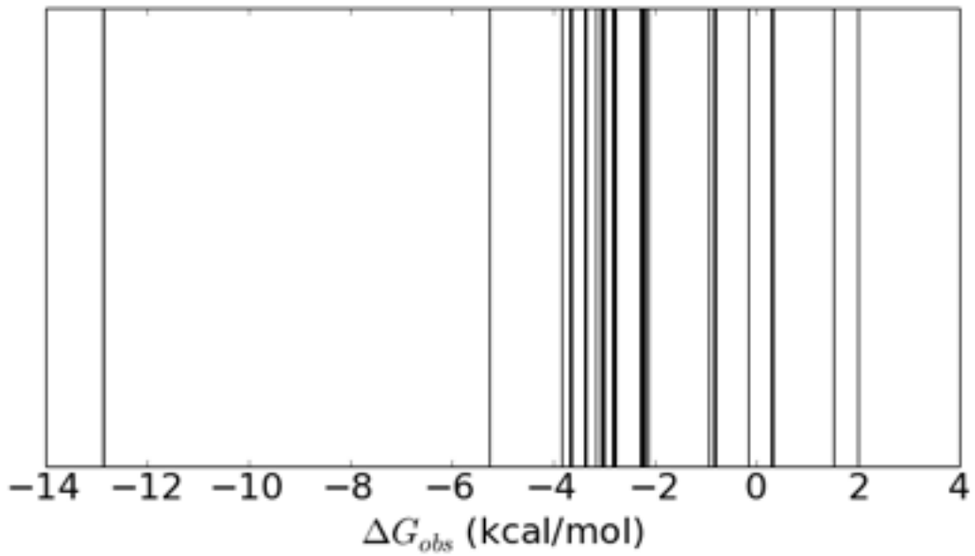
σ **measurement error standard deviation**

This model is simple, but these principles apply to whatever you want

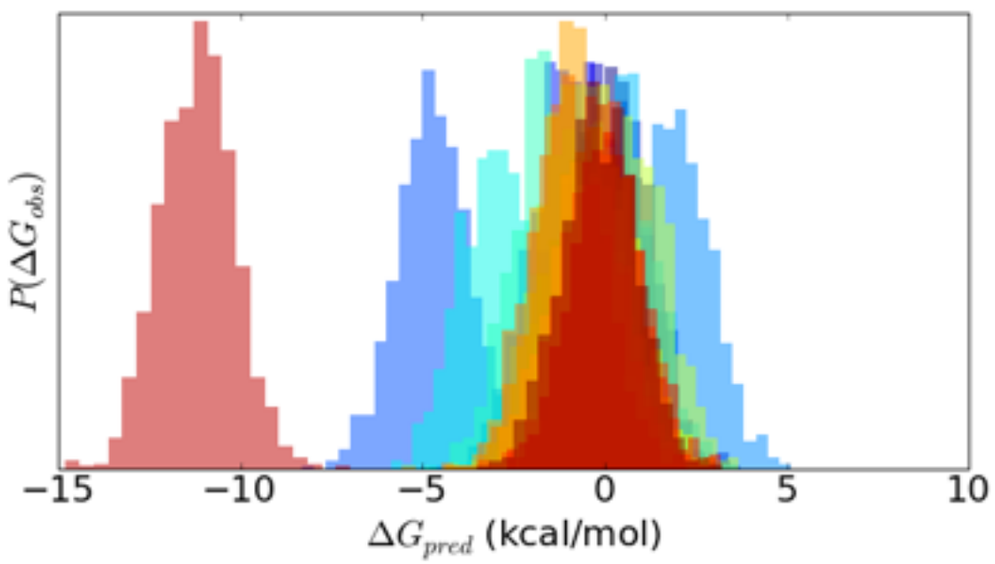
Code at https://github.com/choderalab/cup-xiv

Suppose we have already made some compounds

Measured affinities for 20 compounds

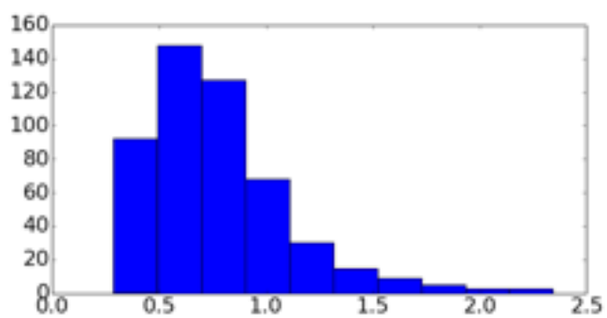


Predictions for next 20 compounds



Some pymc magic to sample models:

```
import pymc

# Create a pymc model
def make_model(x, DGobs_training):
    ntrain = len(x)

    A = pymc.Normal('A', mu=0.0, tau=1.0, size=(M,M))

    log_sigma = pymc.Uniform('log_sigma', lower=-3, upper=0, value=0.0)
    @pymc.deterministic
    def precision(log_sigma=log_sigma):
        return 1.0 / numpy.exp(log_sigma)

    @pymc.deterministic
    def DGmodel(A=A):
        DG = numpy.zeros([ntrain])
        for i in range(ntrain):
            DG[i] = model(x[i], A)
        return DG

    DGobs = pymc.Normal('DGobs', mu=DGmodel, tau=precision, size=[ntrain], observed=True, value=DGobs_training) # observed data

    # Construct dictionary of model variables.
    pymc_model = { 'A' : A, 'log_sigma' : log_sigma, 'precision' : precision, 'DGmodel' : DGmodel, 'DGobs' : DGobs }
    return pymc_model

pymc_model = pymc.Model(make_model(x_training, DGobs_training))

# Sample with MCMC
mcmc = pymc.MCMC(pymc_model, db='ram', name='Sampler', verbose=True)
mcmc.assign_step_methods()
mcmc.sample(iter=10000, burn=5000, thin=10, progress_bar=False)
```



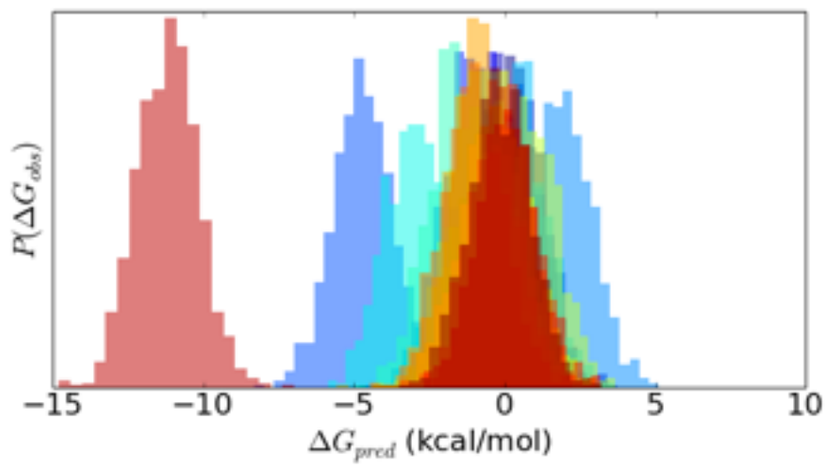
predicted experimental error (actual: 1)

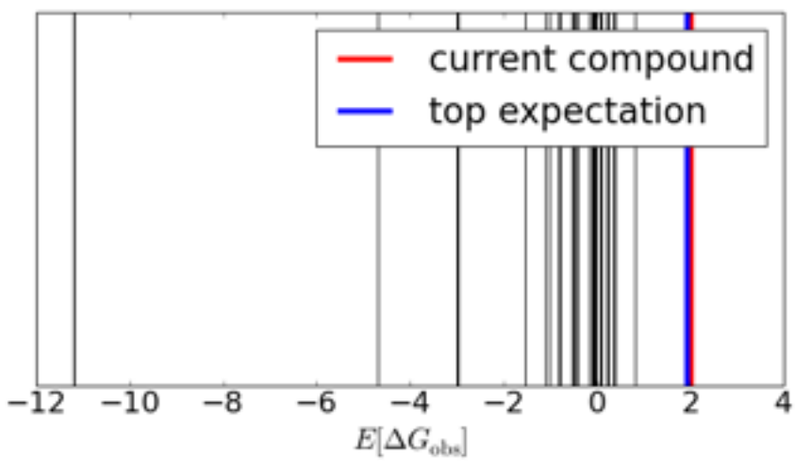
Which ones should we make?

Code at https://github.com/choderalab/cup-xiv

Maximizing the expectation (as in Bayesian bandits)

We could select a compound that maximizes our expected affinity gain (just like Bayesian Bandits)





P(one of 20 compounds has affinity gain) = 0.538
P(top predicted expectation has affinity gain) = 0.361

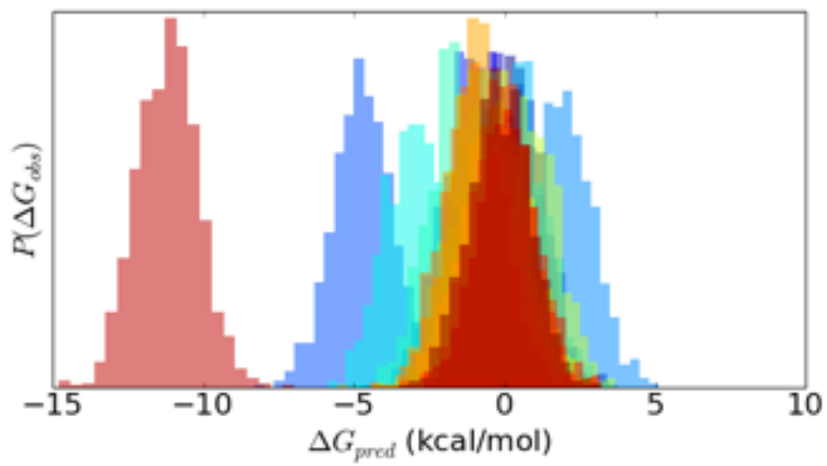
[Not unexpected for a model with 9 parameters fit to data for 10 compounds!]

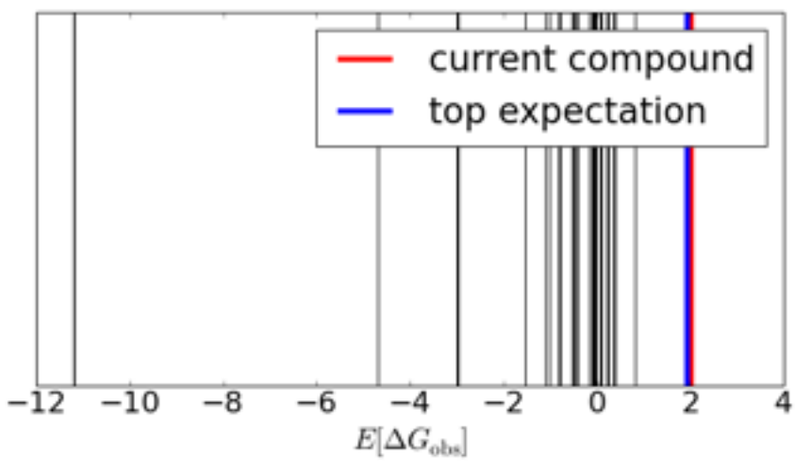
Uh oh. None of the 20 we were given have an expectation higher than the current best compound.

Code at https://github.com/choderalab/cup-xiv

Maximizing the expectation (as in Bayesian bandits)

What if we selected the five compounds that maximize expected affinity gain?





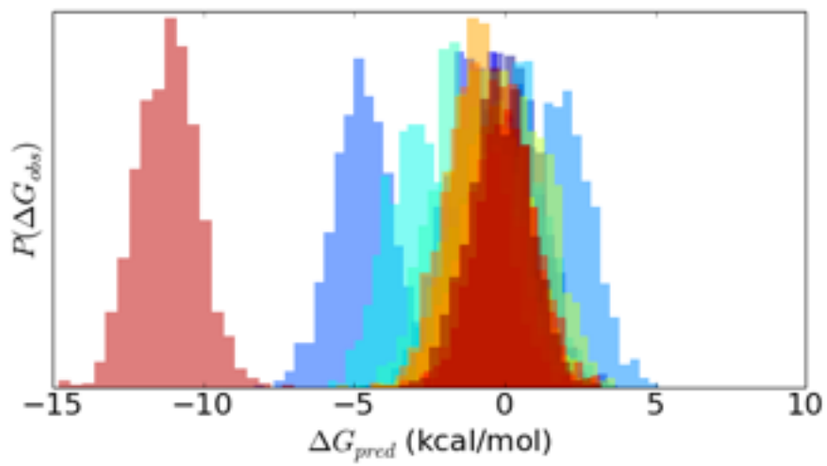
P(one of 20 compounds has affinity gain) = 0.538
P(top predicted expectation has affinity gain) = 0.361
P(affinity gain in top 5) = 0.502

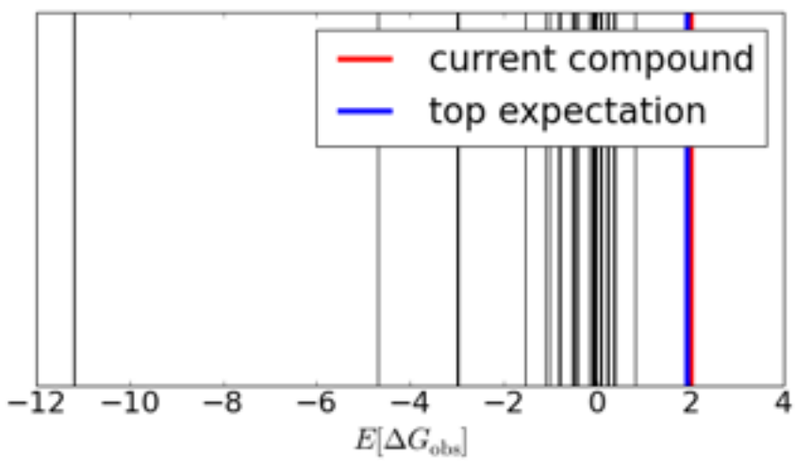
[Doing better!]

Code at https://github.com/choderalab/cup-xiv

Maximizing the expectation (as in Bayesian bandits)

What if we selected the five compounds that minimize probability of failure?





P(one of 20 compounds has affinity gain) = 0.538
P(top predicted expectation has affinity gain) = 0.361
P(affinity gain in top 5) = 0.502
P(affinity gain in "least worst" 5) = 0.539
[A bit better!]

Code at https://github.com/choderalab/cup-xiv

Advanced topics:

Decision-making for the risk-averse

Societal issues often penalize failure much more than rewarding success.

“You only get to be wrong once.”

How can we incorporate risk-aversion into decision-making?

Decision-making for information gain

What experiments will maximize our information gain?

Decision-making with justification

Betting odds can justify expected outcomes

Acknowledgments

Statistical fisticuffs

Ant Nicholls

Kim Branson

ITC and entropy-enthalpy compensation

Kim Branson, Sarah Boyce, Paul Novick, Vijay Pande, David Minh, David Mobley

Telepresents

Johnny 5 and his support staff

(Brian Cole, Craig Bruce, Ben Ellingson)

Slides, references, and iPython tutorials available at http://choderalab.org/cup-xiv