
ForceBalance: Near-term parameter optimization strategy

Lee-Ping Wang, Chemistry Department, UC Davis

Open Force Field Initiative
Consortium meeting, 8 Jan 2019

#forcebalance on Slack

Outline (probably not included in talk)

- **(5 min) What is ForceBalance?**
 - What are some practical challenges of force field parameterization?
 - How does ForceBalance address them?
- **(5 min) How is ForceBalance used?**
 - Installation from conda
 - File and folder structure of a FB calculation
- **(10 min) What is the role of ForceBalance in the Open Force Field effort?**
 - Refer to roadmap slide
 - Preliminary support for SMIRNOFF force field format
 - Show ethanol example (liquid and QM properties)
 - Web interface

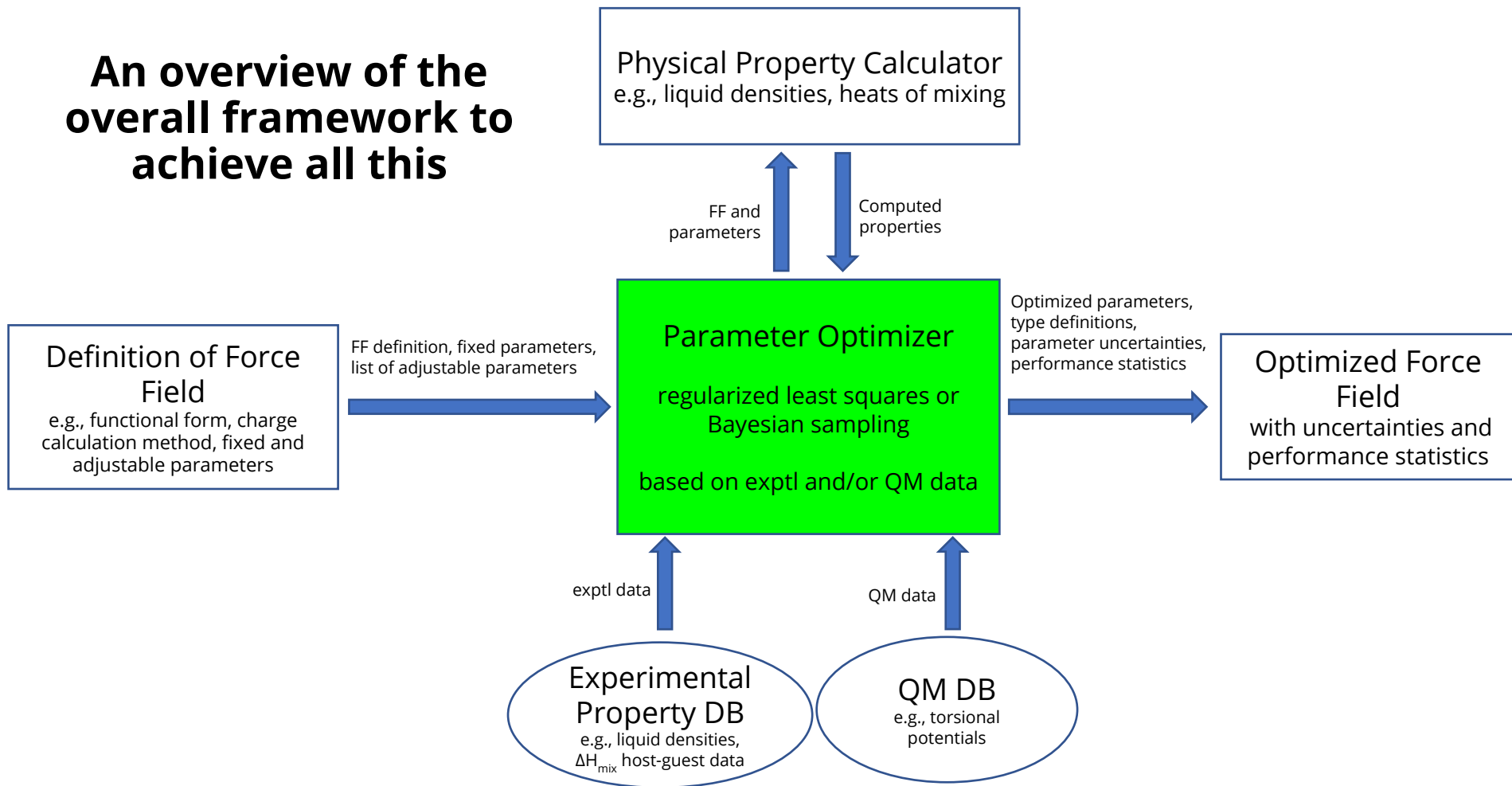
Outline (probably not included in talk)

- (20 min) **What are the plans for ForceBalance in the Open Force Field effort?**
 - Creating the next generations of the SMIRNOFF force field with optimized parameters
 - *Note:* Bayesian optimization will be phased in as it comes online.
 - QM & experimental data will come from data generation and organization subprojects (QCArchive, torsion drives & fragmentation, valence, ThermoML liquid properties)
 - Parameter fitting will initially proceed in a piecewise fashion (subset of parameters to subset of data) followed by fully coupled optimization (all parameters to all data)
 - **Part 1 (3-6 months):** Fitting valence (bond / angle / improper) & torsion parameters to QM. Make references to valence and torsion talks for more details. Tasks specific to FB are:
 - 1.1: Enable the calculation and storage of optimized geometries / vibrational modes for ~1000 input molecules in QCArchive (with Daniel Smith).
 - 1.2: Use vibrational analysis to predict valence degrees of freedom that require scanning along larger displacements, and carry out these scans in QCArchive.
 - 1.3: Create targets in FB by downloading needed QM data from QCArchive.
 - 1.4: New FB target for fitting parameters to reproduce optimized QM geometries.
 - 1.5: Add OpenMM / SMIRNOFF support for existing FB vibrational frequency target.
 - 1.6: Mobley lab will execute the workflow to generate data and produce an initial set of optimized valence parameters.
 - Dr. Yudong Qiu is an ideal candidate to accelerate Task 1 (working in LPW group). He has already contributed significantly to FB and OpenFF efforts without direct funding. Vote to approve his funding will be posted in governing board channel.

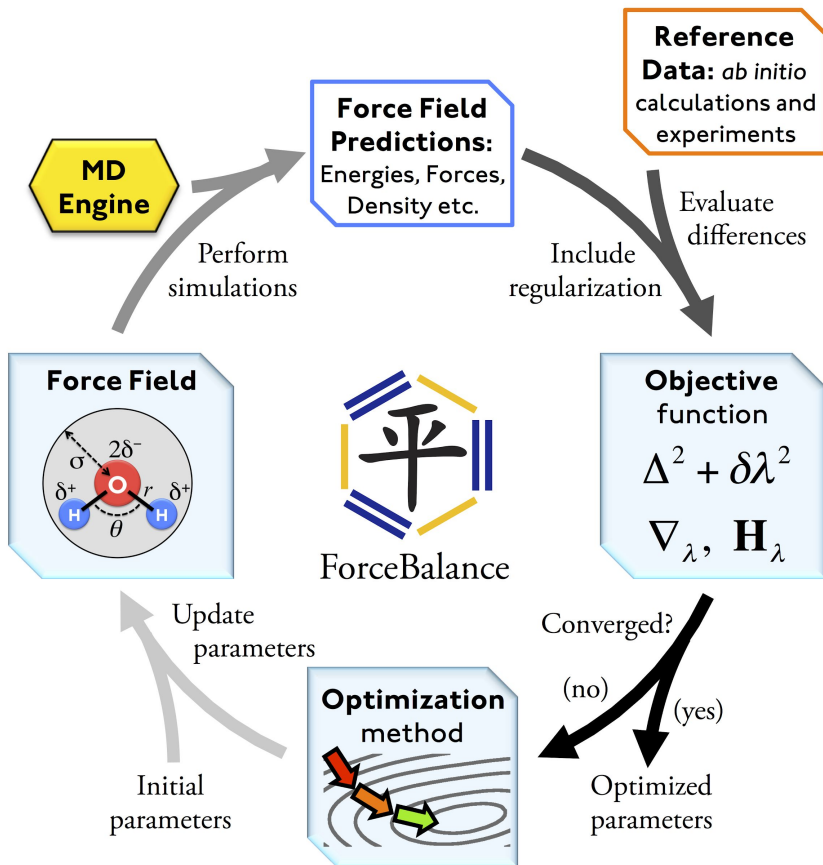
Outline (probably not included in talk)

- **(20 min) What are the plans for ForceBalance in the Open Force Field effort?**
 - **Part 1 continued (3-6 months):** Fitting valence & torsion parameters to QM.
 - Much of this infrastructure is already in place for generating torsional data and torsion parameter fitting (refer to torsion talk.)
 - Valence project is in collaboration with Mobley group (Caitlin Bannan, Victoria Lim, Jessica Maat) & Daniel Smith.
 - Torsion project is in collaboration Chodera group (Chaya Stern) & Daniel Smith.
 - **Part 2 (6-12 months):** Fitting nonbonded parameters to experimental data on ~100 liquids.
 - 2.1: Add a feature to FB that creates liquid property targets by downloading needed experimental data from ThermoML.
 - 2.2: Automatic setup of target files and liquid simulation boxes.
 - 2.3: Execute workflow to produce optimized LJ parameters for liquids.
 - 2.4: Improved FB SMIRNOFF support using OpenForceField toolkit API to modify force field parameters, replacing current approach of writing and reading modified files.
 - 2.5: Replace FB native property calculation code with OpenFF property calculation tool, allowing use of reweighting and surrogate functions to greatly accelerate thermodynamic property estimation.
 - **Part 3 (6+ months):** Fitting of electrostatic parameters, such as AM1-BCC bond charge corrections, to QM and experimental data. One possible route is to integrate FB with AM1-BCC tool being developed (Michael Schauperl & Michael Gilson).
- **(5 min) What are the current challenges and limitations of ForceBalance?**
 - Reliance on predefined parameter types and initial guess; Bayesian optimization a potential solution
 - Difficulty in defining convergence criteria when the objective function contains statistical noise

An overview of the overall framework to achieve all this



ForceBalance is a force field optimization tool



- Python toolkit with a main executable **ForceBalance** for carrying out optimizations.
- Designed for flexibility, FB allows the user to optimize force fields using a wide range of:
 - (1) Functional forms
 - (2) Reference data (QM or expt.)
 - (3) MM simulation software
- Designed for reproducibility, FB enables systematic improvement of models by adding data or physical detail to previous runs.
- Freely available for commercial use via 3-clause BSD license.
- Software distribution comes with 18 example calculations plus all data sets (including those used in published work)

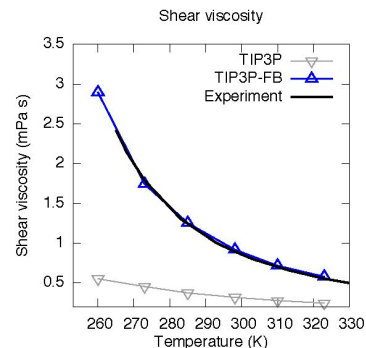
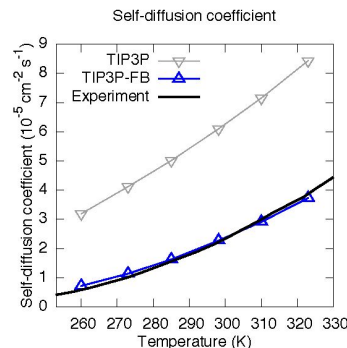
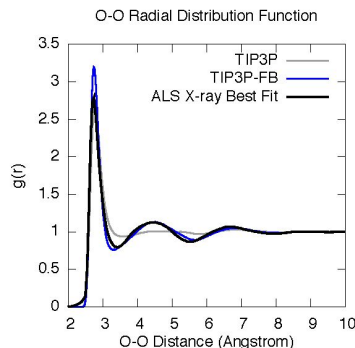
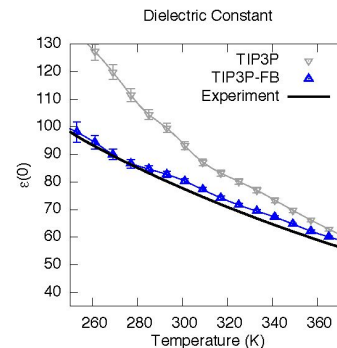
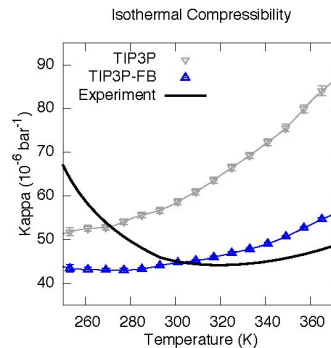
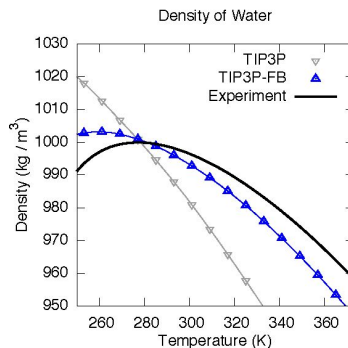
Previous example applications

TIP3P-FB is a water model whose parameters are only a few percent different from TIP3P, but the physical properties of TIP3P-FB are far more accurate.

Top panels: Training properties
(calculated and fitted using FB)

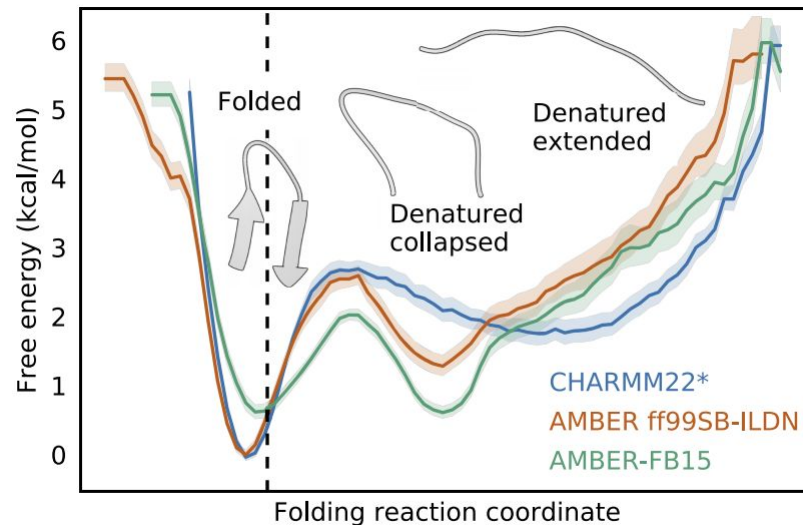
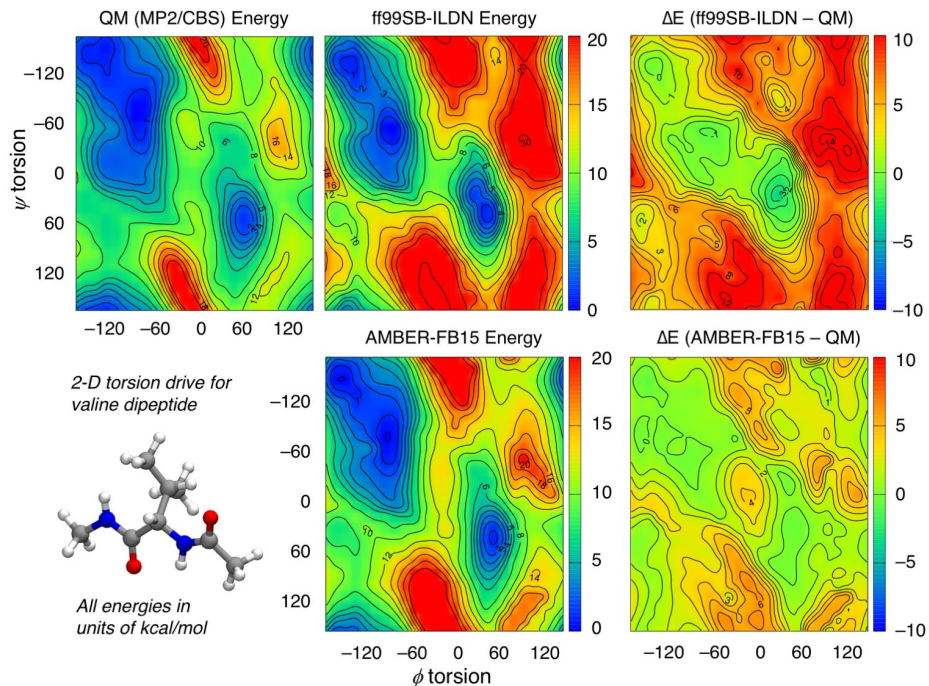
Bottom panels: Validation properties
(calculated outside of FB, not fitted)

Parameter Comparison Table		
	TIP3P	TIP3P-FB
Sigma (Å)	3.1508	3.178
Epsilon (kcal/mol)	0.152	0.1559
Charge (e)	0.417	0.424
Bond (Å)	0.9572	1.0118
Angle (Deg)	104.52	108.15



Previous example applications

AMBER-FB15 is a protein force field with bond, angle, and dihedral parameters optimized to fit high-accuracy QM potential energy surfaces. It predicts temperature dependent properties more accurately than several previously published models.



"The AMBER-FB15 model is 50% folded at the experimental melting temperature, while the AMBER ff99SB-ILDN and CHARMM22* models display an overly populated folded state."

Some force fields built using ForceBalance

- AMBER-FB15 all-atom protein force field (*J. Phys. Chem. B*, 2017)
- GB-FB15 united-atom phospholipid bilayer model (*J. Chem. Theory Comput.*, 2016)
- A series of water models: most of these use the same thermodynamic liquid data set.
 - iAMOEBA, polarizable point dipoles, direct approximation (*J. Phys. Chem. B*, 2013)
 - Referred to as the most accurate polarizable water model in a broad-ranging literature survey located at http://www1.lsbu.ac.uk/water/water_models.html
 - AMOEBA14, mutual approximation (*J. Phys. Chem. B*, 2015)
 - uAMOEBA, united atom polarizable point dipoles (*J. Chem. Phys.*, 2015)
 - TIP3P-FB and TIP4P-FB, rigid fixed-charge (*J. Phys. Chem. Lett.*, 2014)
 - Buckingham models fitted to RDF data (*J. Chem. Inf. Model.*, 2018, with David Huggins)
- Polarizable nanoporous graphene model (*J. Chem. Theory Comput.* 2018, Yudong Qiu)
- Organochlorine compounds with σ -hole effect (*J. Phys. Chem. B*, 2014, with Pengyu Ren)
- Coarse-grained force field fitted to hydration free energies (*submitted*, with Jon Essex)
- Atom-based grids in tensor hypercontraction MP2 method (*J. Chem. Theory Comput.*, 2015)
 - ForceBalance can be easily modified to optimize parameters outside of MM simulations.

Data set characteristics

Water data set. **QM** and **gas phase** data was used for polarizable water models. **Liquid data** has 34 thermodynamic phase points spanning 249 – 373 K and 1 – 8000 bar.

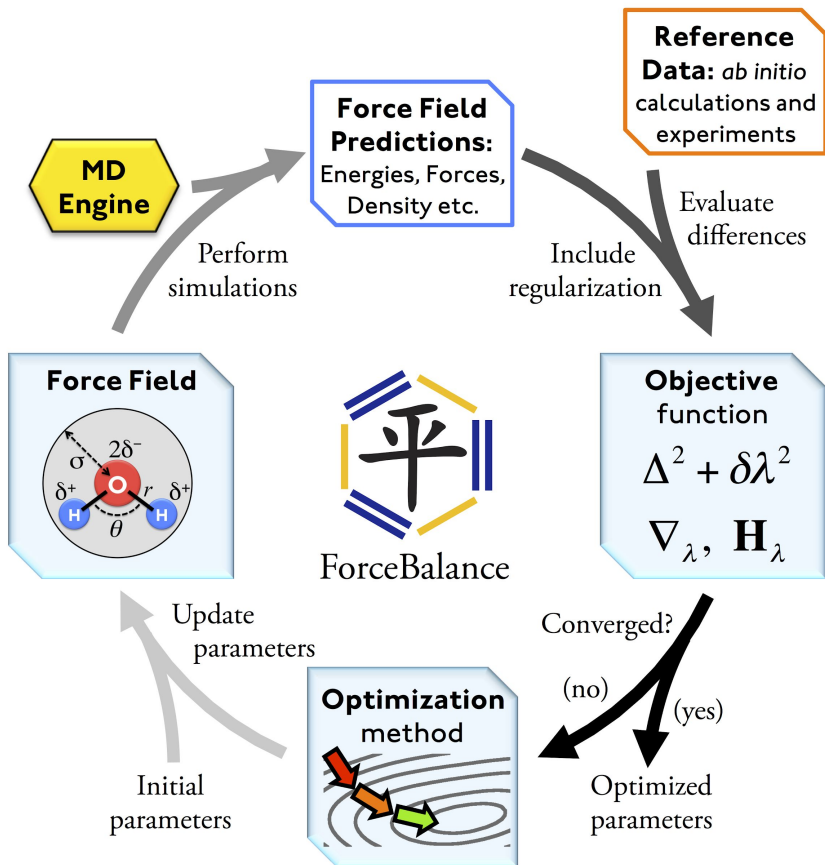
Reference Data	Scaling Factor
Density ρ	2 kg m ⁻³
Heat of Vaporization ΔH_{vap}	5 kJ mol ⁻¹
Thermal Expansion Coefficient α	10 ⁻⁴ K ⁻¹
Isothermal Compressibility κ_T	10 ⁻⁵ bar ⁻¹
Isobaric Heat Capacity c_p	2 kJ mol ⁻¹ K ⁻¹
Dielectric Constant $\epsilon(0)$	2
Gas Phase Dipole	0.2 Debye
Gas Phase Quadrupole	1.0 Debye Å
Gas Phase Vibrational Modes	30 cm ⁻¹
Water Dimer CCSD(T)/CBS Binding Energy, RMSD	1.0 kcal mol ⁻¹ , 0.1 Å
21 Small Gas Phase Clusters (size 3-6) CCSD(T)/CBS Binding Energy, RMSD	10.0 kcal mol ⁻¹ , 2.0 Å
18 Large Gas Phase Clusters (size 8-20) MP2/CBS Binding Energy, RMSD	100.0 kcal mol ⁻¹ , 2.0 Å
42,000 MP2/aug-cc-pVTZ Potential Energies and Atomistic Forces	stdev(E_{QM}), RMS($ \mathbf{F}_{\text{QM}} $)

Protein data set (all QM) for AMBER-FB15 protein force field (~1400 parameters):

reference data	no. of calcs.
energy, gradients of 26 amino acids over (φ, ψ) (incl. ASH, CYM, GLH, HIE, HIP, LYN)	14,971
energy, gradients of 21 amino acids over (χ_1, χ_2) (excluding ALA, CYM, GLY, PRO, VAL)	12,093
energy, gradients of CYM, VAL over (φ, χ_1)	1,151
vibrational frequencies and eigenvectors for 20 amino acids	20
energy, gradients of MM-optimized structures	1,060

All water and protein data included in the FB software distribution.

Optimization workflow



- *Left*: Starting from a labeled force field file with initial guesses, simulations are performed by making automated calls to MD engines.
- *Top*: Properties are calculated and compared to provided reference data to compute objective function.
 - Property derivatives are evaluated by finite difference by writing force field files with perturbed parameter values, reading them back in, and re-evaluating the properties.
- *Right*: Objective function is calculated with regularization included.
- *Bottom*: Optimization step updates mathematical parameters and produces new force field file. Repeat until convergence.

Theory of force field parameter updates

Force field parameters vary across many orders of magnitude and may obey complex functional relationships and constraints.

In FB, the optimization algorithm sees an array of **mathematical parameters** that are well-behaved, i.e. are fully unconstrained and are on order 1.

Physical parameters are related to mathematical parameters by shifting and scaling:

$$\text{Physical parameters} = \text{Initial values} + \text{Rescaling matrix} \times \text{Mathematical parameters}$$

$$\mathbf{k}_{\text{phys}} = \mathbf{k}_{\text{phys}}^{(0)} + \mathbf{T} \mathbf{k}_{\text{math}}$$

Rescaling matrix consists of **prior widths** on diagonal representing the size of expected changes over the optimization (or over parameters of the same type).

Typically, one prior width should be specified for each parameter type (fewer than 10 independent user-specified values).

$$\mathbf{T} = \begin{pmatrix} 0.01 \text{ nm} & 0 & 0 & 0 \\ 0 & 10^5 \frac{\text{kJ}}{\text{mol nm}} & 0 & 0 \\ 0 & 0 & 10^\circ & 0 \\ 0 & 0 & 0 & 10^2 \frac{\text{kJ}}{\text{mol rad}} \end{pmatrix}$$

Theory of force field parameter updates

The rescaling matrix **T** is almost always diagonal; off-diagonal could be used to constrain net charges on molecules to stay constant.

More generally, ***evaluated parameters*** may be defined as any mathematical function of physical parameters:

Full set of
parameters
comprises:

Physical
parameters,

Evaluated
parameters,

and deeper evaluated
parameters if any...

$$\mathbf{k}_{\text{full}} = \mathbf{k}_{\text{phys}} \oplus \mathbf{k}_{\text{eval}}^{(0)} \left(\mathbf{k}_{\text{phys}} \right) \oplus \mathbf{k}_{\text{eval}}^{(1)} \left(\mathbf{k}_{\text{phys}} ; \mathbf{k}_{\text{eval}}^{(0)} \right) \oplus \dots$$

“Physical” & “evaluated” parameters may be used as scratch variables not to be read by the MM software. Thus, parameters that are actually used by the MM software can be defined such that they obey almost any desired mathematical relationship - such as summing up to a constant, restricted to within a range, or obeying a geometric / trigonometric relationship.

Theory of objective function

The objective function is a weighted sum of least-squares contributions called *targets* plus regularization:

$$L_{\text{tot}}(\mathbf{k}_{\text{math}}) = \sum_{i \in \text{targets}} w_i L_i(\mathbf{k}_{\text{math}}) + w_{\text{reg}} |\mathbf{k}_{\text{math}}|^2$$

Objective function
Weighted sum over targets
Regularization

user-specified w , unity usually sufficient

Each target is a weighted sum of least-squares contributions for one or more properties:

$$L_i(\mathbf{k}_{\text{math}}) = \sum_{j \in \text{properties}} w_{ij} L_{ij}(\mathbf{k}_{\text{math}})$$

User-specified weights for properties
(unity usually sufficient)

Each property is a weighted and normalized sum over individual data points:

$$L_{ij}(\mathbf{k}_{\text{math}}) = \frac{1}{d_{ij}^2} \frac{\sum_{p \in \text{points}} w_{ijp} \left| y_{ijp}(\mathbf{k}_{\text{math}}) - y_{ijp}^{(\text{ref})} \right|^2}{\sum_{p \in \text{points}} w_{ijp}}$$

Overall normalization to remove units
Weighted, normalized sum over data points
(uniform or automatic weights)

Theory of optimization algorithm

The matrix of second derivatives (*Hessian*) of a least-squares objective function can be estimated if the first derivatives of residuals are known (Gauss-Newton approximation):

$$H_{pq}[\mathbf{k}_{\text{math}}] = \frac{\partial^2}{\partial k_p \partial k_q} \left| y(\mathbf{k}_{\text{math}}) - y^{(\text{ref})} \right|^2 = 2 \frac{\partial y}{\partial k_q} \frac{\partial y}{\partial k_p} + \underbrace{2 \frac{\partial^2 y}{\partial k_p \partial k_q}}_{\text{approximate as zero}}$$

This enables highly efficient quasi-Newton optimization algorithms to be used.

$$\mathbf{k}_{\text{math}}^{(i+1)} = \mathbf{k}_{\text{math}} + \left(\mathbf{H}_{\text{approx}}[\mathbf{k}_{\text{math}}] + \lambda \mathbf{I} \right)^{-1}$$

The λ parameter is used to restrict the optimization step to lie within a trust radius (which is adjusted on-the-fly based on step quality), or it can be used in line-search minimization to determine the next step.

FB implements BFGS Hessian updating algorithm as an alternative, less efficient approach.

Liquid thermodynamic property target

- **Liquid** target allows simultaneous fitting of up to six thermodynamic properties (density, heat of vaporization, thermal expansion coefficient, isothermal compressibility, isobaric heat capacity, and dielectric constant) over a range of temperatures and pressures.
- One NPT simulation is run per thermodynamic phase point.
- From saved trajectory frames, calculate potential energy derivatives using finite-difference
- Thermodynamic fluctuation formula enables computation of property derivatives w/r.t. parameters using potential energy derivatives, without having to run separate simulations for each parameter.

Ensemble-averaged density

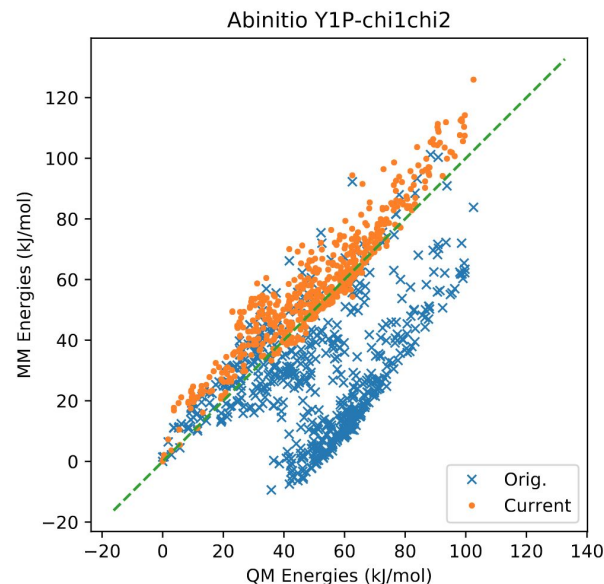
$$\rho = \left\langle \frac{M}{V} \right\rangle = \frac{1}{Q_{\text{NPT}}} \int e^{-\beta(E(\mathbf{r};k)+PV)} \frac{M}{V}$$

Ensemble-averaged density
derivative with respect to parameter

$$\frac{\partial \rho}{\partial k} = \left\langle -\beta \frac{\partial E}{\partial k} \frac{M}{V} \right\rangle - \left\langle -\beta \frac{\partial E}{\partial k} \right\rangle \left\langle \frac{M}{V} \right\rangle$$

Ab initio energy and force target

- **AbInitio** target allows fitting of single-point energies and atomistic forces to pre-computed QM data for a set of configurations.
- Weighting scheme emphasizes configurations with low QM energy (because they are more thermodynamically favorable).
- Additional emphasis is placed on avoiding MM energy lower than QM energy (because that would cause incorrect structures being sampled at equilibrium).
 - In this scheme, energy is referenced to the structure with lowest QM energy.
- Figure at right shows a scatter plot of QM vs. MM energies for a 2D torsion drive on the side chain of phosphorylated tyrosine (blue = initial parameters; orange = after 2 optimization steps)



Plot courtesy John Stoppelman (GA Tech)

A wide range of targets and software are supported

- In addition to [Liquid](#) and [AbInitio](#) target, various other targets are supported e.g. binding energies, vibrational frequencies, gas-phase multipole moments, surface tension (Yudong Qiu), lipid membrane properties (Keri McKiernan), implicit solvent hydration free energies
- Multiple simulation software packages (**engines**) and force field formats are supported including AMBER, OpenMM, GROMACS, TINKER
 - *SMIRNOFF force field support added as of December 2018.* (OpenEye commercial toolkit is required until Open Force Field toolkit implements RDKit support).
- Most codes are target-specific or engine-specific, such that minimal implementation is needed to fill the matrix of possible target / engine combinations.
- Work Queue library supports computationally intensive optimization jobs by evaluating targets or individual simulations on remote computer resources.
 - Work Queue is GPL v2, but this optional component does not “infect” ForceBalance with copyleft and the author of Work Queue is enthusiastic about FB being BSD-licensed and using it.

Installation using conda package manager

```
# Download and install Miniconda.
# Create Python environment and configure Conda channels
conda create --name fb_smirnoff_py36 python=3.6
source activate fb_smirnoff_py36
conda config --add channels omnia --add channels conda-forge

# Install ForceBalance from omnia channel
conda install forcebalance

# To use FB with SMIRNOFF, we need OpenMM and the Open Force Field toolkit
module load cuda/9.2 # Load CUDA environment for OpenMM (may vary on your system)
conda install openmm # Install OpenMM from omnia channel

# Commercial OpenEye toolkit is currently a prerequisite of using SMIRNOFF force field.
# Should support open-source RDKit within ~1 month.
conda install -c openeye openeye-toolkits

# Install Open Force Field toolkit (for setting up SMIRNOFF simulations)
conda install openforcefield

# Obtain latest ForceBalance source code from GitHub with examples
# Note: You may install the latest source code without waiting for a new release.
mkdir $HOME/src ; cd $HOME/src
git clone git@github.com:leeping/forcebalance.git

# Run example parameter optimization calculation
cd $HOME/src/forcebalance/studies/017_smirnoff_ethanol
ForceBalance optimize.in
```

Setting up a calculation:

Selecting parameters in force field file

Black: Existing SMIRNOFF 0.1 force field format; <Atom .../> is a parameter line

Blue: Parameter selections for ForceBalance to optimize parameters

Red: Parameters that ForceBalance will optimize based on parameter selections

For readability, some of the XML tags are shown on multiple lines

```
<NonbondedForce coulomb14scale="0.833333" lj14scale="0.5" sigma_unit="angstroms"
  epsilon_unit="kilocalories_per_mole">

  <Atom smirks="#1:1" epsilon="0.0157" rmin_half="0.6000"/>

  <Atom smirks="#1:1-[#6X4]"
    epsilon="0.0157" rmin_half="1.4870" parameterize="rmin_half, epsilon"/>

  <Atom smirks="#1:1-[#6X4]-[#7,#8,#9,#16,#17,#35]"
    epsilon="0.0157" rmin_half="1.3870" parameterize="rmin_half, epsilon"/>

</NonbondedForce>
```

Setting up a calculation: File and directory structure

```
├── forcefield # Contains all files containing parameters to be optimized
│   └── smirnoff99Frosst.offxml
├── targets    # Contains QM / experimental data and associated simulation files
│   ├── ethanol-liquid # Density and heat of vaporization of liquid ethanol
│   │   ├── data.csv
│   │   ├── ETH.mol2
│   │   ├── gas.pdb
│   │   └── liquid.pdb
│   └── ethanol-torsiondrive # Ethanol torsional potential energy surface
│       ├── ETH.mol2
│       ├── gas.pdb
│       ├── qdata.txt
│       └── scan.xyz
└── optimize.in # Input file with optimization specifications
```

- Once targets are set up, they typically do not need to be modified.
- To select which parameters are being optimized, edit parameter selections in force field files.
- Other fine-tuning of optimizations (such as regularization strength, choice of which targets to use, relative weights, simulation length) are set in the input file.

Output and optimization progress

Starting SMIRNOFF parameters

```
#=====#
#|          ethanol-liquid Density (kg m^-3)          |#
#| Temperature Pressure Reference Calculated +- Stdev   Delta   Weight   Term   |#
#=====#
      298.15      1.0 atm   785.000      798.134 +- 0.642   13.134   0.50000   0.09584
      320.03      1.0 atm   766.000      775.046 +- 0.723    9.046   0.50000   0.04546
#=====#
#|          ethanol-liquid Enthalpy of Vaporization (kJ mol^-1)          |#
#=====#
      298.15      1.0 atm   42.300      35.809 +- 0.187   -6.491   0.50000   2.34094
      320.03      1.0 atm   41.000      34.183 +- 0.217   -6.817   0.50000   2.58147
#=====#
#| Target: ethanol-torsiondrive Type: AbInitio_SMIRNOFF Objective = 1.26640e+00 |#
#|          Difference Denominator Percent          |#
#| Observable      (Calc-Ref)      RMS (Ref)  Difference      Term   x Wt =  Contrib. |#
#=====#
      Energy (kJ/mol)      2.0729      1.8420   112.5345%      1.2664      1.000      1.2664

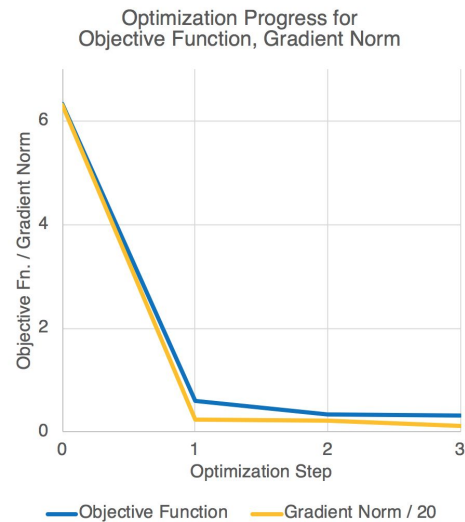
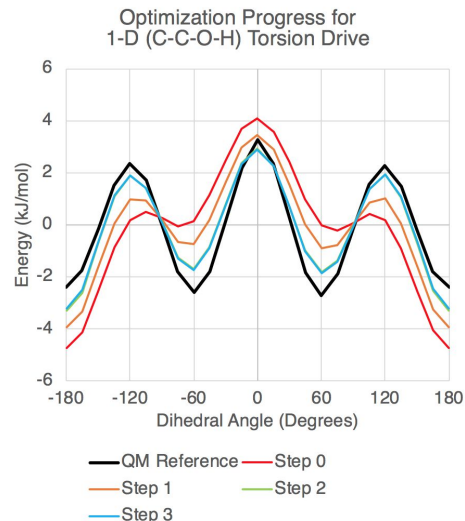
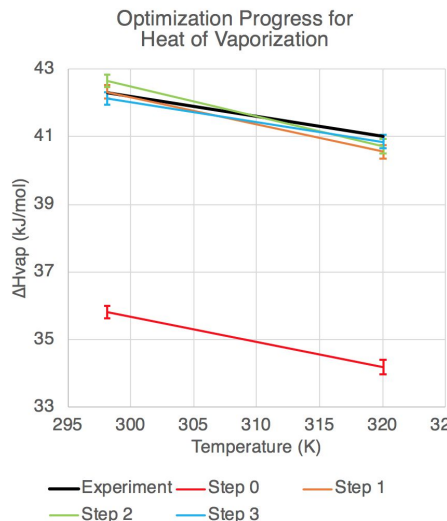
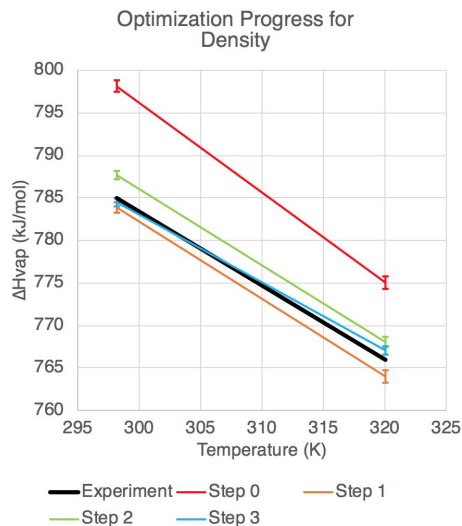
Step      |k|      |dk|      |grad|      -=X2=-      Delta(X2)      StepQual
  0      0.000e+00      0.000e+00      1.260e+02      6.33011e+00      -0.000e+00      0.000
```

Output and optimization progress

Starting SMIRNOFF parameters; After 1 optimization step

#=====								
#	ethanol-liquid Density (kg m ⁻³)							
#	Temperature	Pressure	Reference	Calculated	+ - Stdev	Delta	Weight	Term
#	#=====							
	298.15	1.0 atm	785.000	798.134	+ - 0.642	13.134	0.50000	0.09584
	320.03	1.0 atm	766.000	775.046	+ - 0.723	9.046	0.50000	0.04546
	298.15	1.0 atm	785.000	783.856	+ - 0.570	-1.144	0.50000	0.00073
	320.03	1.0 atm	766.000	764.036	+ - 0.704	-1.964	0.50000	0.00214
#	#=====							
#	ethanol-liquid Enthalpy of Vaporization (kJ mol ⁻¹)							
#	#=====							
	298.15	1.0 atm	42.300	35.809	+ - 0.187	-6.491	0.50000	2.34094
	320.03	1.0 atm	41.000	34.183	+ - 0.217	-6.817	0.50000	2.58147
	298.15	1.0 atm	42.300	42.326	+ - 0.197	0.026	0.50000	0.00004
	320.03	1.0 atm	41.000	40.563	+ - 0.197	-0.437	0.50000	0.01063
#	#=====							
#	Target: ethanol-torsiondrive Type: AbInitio_SMIRNOFF Objective = 1.26640e+00							
#	#=====							
#	Observable	Difference (Calc-Ref)	Denominator RMS (Ref)	Percent Difference	Term	x Wt =	Contrib.	
#	#=====							
	Energy (kJ/mol)	2.0729	1.8420	112.5345%	1.2664	1.000	1.2664	
	Energy (kJ/mol)	1.3231	1.8420	71.8310%	0.5160	1.000	0.5160	
Step	k	dk	grad	--X2--	Delta(X2)	StepQual		
0	0.000e+00	0.000e+00	1.260e+02	6.33011e+00	-0.000e+00	0.000		
1	2.500e-01	2.500e-01	4.889e+00	5.92008e-01	-5.738e+00	1.002		

Output and optimization progress



- Typically, convergence is obtained in <10 optimization steps
- Quality of results for liquid properties depends on simulation length. Here we ran 1.0 ns NPT simulations.
- Statistical noise in liquid properties causes fluctuations in objective function and defining appropriate convergence criteria is still a challenge. Here our convergence criterion for the objective function decrease is set to 0.1.

Optimization results: File and directory structure

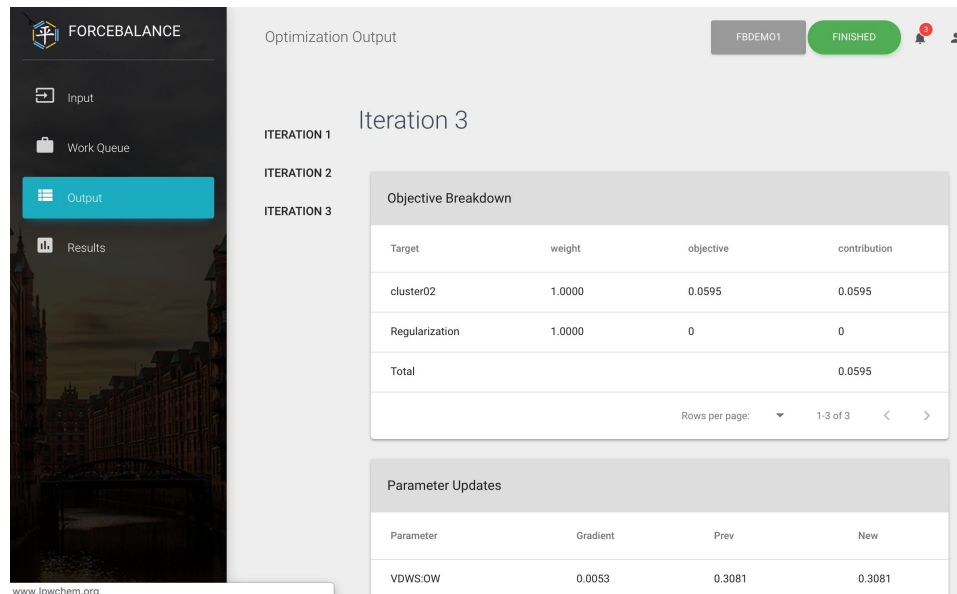
```
— forcefield # Contains all files containing parameters to be optimized
  └─ smirnoff99Frosst.offxml
— targets    # Contains QM / experimental data and associated simulation files
  └─ ethanol-liquid # Density and heat of vaporization of liquid ethanol
    └─ data.csv
    └─ ETH.mol2
    └─ gas.pdb
    └─ liquid.pdb
  └─ ethanol-torsiondrive # Ethanol torsional potential energy surface
    └─ ETH.mol2
    └─ gas.pdb
    └─ qdata.txt
    └─ scan.xyz
— optimize.in # Input file with optimization specifications
```

Files generated during ForceBalance optimization are below this line

```
— optimize.tmp # Working folder containing intermediate calculations and results
  └─ ethanol-liquid # Organization by target, then by iteration number
    └─ iter_0000
      └─ (calculation files and folders found here)
— result      # Force field files with parameters replaced by optimized values
  └─ optimize
    └─ smirnoff99Frosst.offxml
— optimize.out # Output file summarizing optimization progress
— optimize.sav # Checkpoint file for continuing optimization from latest point
```

Web interface for interactive ForceBalance calculations

- Written by Dr. Yudong Qiu with support from MolSSI Fellowship
- Use it as a GUI for FB running locally, or set up a Web server
- Interactive wizard for setting up of targets
- Real-time feedback and visualization of optimization results
- Same file and directory structure as CLI for reproducible runs
- Basic functionality works, still need to support the full range of targets in FB.



The screenshot displays the FORCEBALANCE web interface. On the left is a dark sidebar with navigation links: Input, Work Queue, Output (highlighted in blue), and Results. The main area is titled 'Optimization Output' and shows 'Iteration 3' selected. Below this, there is an 'Objective Breakdown' table and a 'Parameter Updates' table. The interface also includes a status bar at the top right with 'FBDEMO1', a green 'FINISHED' button, and notification icons.

FORCEBALANCE

Input

Work Queue

Output

Results

Optimization Output

FBDEMO1 FINISHED

Iteration 3

ITERATION 1

ITERATION 2

ITERATION 3

Objective Breakdown

Target	weight	objective	contribution
cluster02	1.0000	0.0595	0.0595
Regularization	1.0000	0	0
Total			0.0595

Rows per page: 1-3 of 3

Parameter Updates

Parameter	Gradient	Prev	New
VDWS:OW	0.0053	0.3081	0.3081

www.lawchem.org

Plans for ForceBalance in the Open Force Field effort

- Creating the next generations of the SMIRNOFF force field with optimized parameters
 - *Note:* Bayesian optimization (with **Chodera lab**) will be phased in as it comes online.
- QM & experimental data will come from data generation and organization subprojects (QCArchive, torsion drives & fragmentation, valence, ThermoML liquid properties)
- Parameter fitting will initially proceed in a piecewise fashion (subset of parameters to subset of data) followed by fully coupled optimization (all parameters to all data)
- **Part 1 (3-6 months):** Fitting valence (bond / angle / improper) & torsion parameters to QM. Make references to valence and torsion talks for more details. Tasks specific to FB are:
 - 1.1: Enable the calculation and storage of optimized geometries / vibrational modes for ~1000 input molecules in QCArchive (with **Daniel Smith**).
 - 1.2: Scan the potential energy along more deformable degrees of freedom such as sp² nitrogen pyramidalization (with **Mobley lab**), and support this in QCArchive.
 - 1.3: Create targets in FB by downloading needed QM data from QCArchive.
 - 1.4: New FB target for fitting parameters to reproduce optimized QM geometries.
 - 1.5: Add OpenMM / SMIRNOFF support for FB vibrational frequency target.
 - 1.6: Execute workflow to generate data and optimize valence parameters; to be carried out in collaboration with Mobley lab members (**Caitlin Bannan, Victoria Lim, Jessica Maat**).
 - **Dr. Yudong Qiu** is an ideal candidate to accelerate coding tasks (in Wang group). He has already contributed significantly to FB and OpenFF efforts without being directly funded by OpenFF. Discussion to approve his funding will be posted in #governing-board channel.

Plans for ForceBalance in the Open Force Field effort

- Much progress and still much to do (outside of FB) for torsional data and torsion parameter fitting.
 - Torsion project is in collaboration with Chodera lab (**Chaya Stern**) & Daniel Smith.
- **Part 2 (6-12 months):** Fitting Lennard-Jones parameters to experimental data on ~100 liquids.
 - 2.1: Add a feature to FB that creates liquid property targets by downloading needed experimental data from ThermoML (**Michael Shirts** & **Kenneth Kroenlein**).
 - 2.2: Automatic setup of target files and liquid simulation boxes.
 - 2.3: Execute workflow to produce optimized LJ parameters for liquids.
 - 2.4: Improved FB SMIRNOFF support using OpenForceField toolkit API (**Jeffrey Wagner**) to modify force field parameters, replacing current approach of writing and reading modified files.
 - 2.5: Replace FB native property calculation code with OpenFF property calculation tool (Shirts & Chodera labs + **Simon Boothroyd**), allowing use of reweighting and surrogate functions to greatly accelerate thermodynamic property estimation.
- **Part 3 (6+ months):** Fitting of electrostatic parameters, such as AM1-BCC bond charge corrections, to QM and experimental data. One possible route is to integrate FB with AM1-BCC tool being developed (**Michael Schaeperl**, **Michael Gilson**, with input from **Paul Nerenberg**).
- **(5 min) What are the current challenges and limitations of ForceBalance?**
 - Reliance on predefined parameter types, initial guess, prior widths, and target weights.
 - Bayesian optimization is a possible long-term solution.
 - Difficulty in defining convergence criteria when the objective function contains statistical noise.