

JuPOETs: A Constrained Multiobjective Optimization Approach to Estimate Biochemical Model Ensembles in the Julia Programming Language

David Bassen¹, Michael Vilkhovoy, Mason Minot, Jonathan T Butcher¹ and Jeffrey D. Varner*

School of Chemical and Biomolecular Engineering and

¹School of Biomedical Engineering, Cornell University, Ithaca NY 14853

Running Title: Multiobjective model ensemble generation

To be submitted: *BMC Systems Biology*

*Corresponding author:

Jeffrey D. Varner,

Professor, School of Chemical and Biomolecular Engineering,

244 Olin Hall, Cornell University, Ithaca NY, 14853

Email: jdv27@cornell.edu

Phone: (607) 255 - 4258

Fax: (607) 255 - 9166

Abstract

Ensemble modeling is a well established approach for obtaining robust predictions and for simulating course grained population behavior in deterministic mathematical models. In this study, we present a multiobjective based technique to estimate model ensembles, the Pareto Optimal Ensemble Technique in the Julia programming language (JuPOETs). JuPOETs integrates simulated annealing with Pareto optimality to estimate parameter ensembles on or near the optimal tradeoff surface between competing training objectives. We demonstrated JuPOETs on a suite of multiobjective problems, including test functions with parameter bounds and system constraints as well as for the identification of a proof-of-concept biochemical model with four conflicting training objectives. JuPOETs identified optimal or near optimal solutions approximately six-fold faster than a corresponding implementation in Octave for the suite of test functions. For the proof-of-concept biochemical model, JuPOETs produced an ensemble of parameters that gave both the mean of the training data for conflicting data sets, while simultaneously estimating parameter sets that performed well on each of the individual objective functions. JuPOETs can be adapted to solve many problem types, including mixed binary and continuous variable types, bilevel optimization problems and constrained problems without altering the base algorithm. JuPOETs can be installed using the Julia package manager from the JuPOETs GitHub repository at <https://github.com/varnerlab/POETs.jl>.

1 **Background**

2 Ensemble modeling is a well established approach for obtaining robust predictions and
3 course grained population behavior in large deterministic mathematical models. It is of-
4 ten not possible to uniquely identify all the parameters in biochemical models, even when
5 given extensive training data [1]. Thus, despite significant advances in standardizing bio-
6 chemical model identification [2], the problem of estimating model parameters from exper-
7 imental data remains challenging. Ensemble approaches address parameter uncertainty
8 in systems biology and other fields like weather prediction [3–6] by using parameter fam-
9 ilies instead of single best-fit parameter sets. Parameter families can be selected based
10 upon simulation error, along with other criterion such as diversity or steady-state perfor-
11 mance. Simulations using parameter ensembles can estimate confidence intervals on
12 model variables, and robustly constrain model predictions, despite having many poorly
13 constrained parameters [7, 8]. There are many techniques to generate parameter en-
14 sembles. Battogtokh et al., Brown et al., and later Tasseff et al. generated experimentally
15 constrained parameter ensembles using a Metropolis-type random walk [3, 5, 9, 10]. Liao
16 and coworkers developed methods that generate ensembles that all approach the same
17 steady-state, for example one determined by fluxomics measurements [11]. They have
18 used this approach for model reduction [12], strain engineering [13, 14] and to study the
19 robustness of non-native pathways and network failure [15]. Maranas and coworkers
20 have also applied this method to develop a comprehensive kinetic model of bacterial cen-
21 tral carbon metabolism, including mutant data [16]. We and others have used ensemble
22 approaches, generated using both sampling and optimization techniques, to robustly sim-
23 ulate a wide variety of signal transduction processes [9, 10, 17–19], neutrophil trafficking
24 in sepsis [20], patient specific coagulation behavior [21] and to capture cell to cell varia-
25 tion [22]. Thus, ensemble approaches are widely used to robustly simulate a variety of
26 biochemical systems.

27 Identification of biochemical models with hundreds or even thousands of states and
28 parameters may not be tractable as a single objective optimization problem. Further, large
29 models require significant training data perhaps taken from diverse sources, for example
30 different laboratories or cell-lines. These data are often heterogenous, and contain intrinsic
31 conflicts that complicate parameter estimation. Parameter ensembles which optimally
32 balance tradeoffs between submodels and conflicts in training data can lead to robust
33 model performance. Multiobjective optimization is an ensemble generation technique that
34 naturally balances conflicting training data. Previously, we developed the Pareto Optimal
35 Ensemble Technique (POETs) algorithm to address the challenge of competing or conflicting
36 objectives. POETs, which integrates simulated annealing (SA) and multiobjective
37 optimization through the notion of Pareto rank, estimates parameter ensembles which
38 optimally trade-off between competing (and potentially conflicting) experimental objectives
39 [23]. However, the previous implementation of POETs, in the Octave programming
40 language [24], suffered from poor performance and was not configurable. For example,
41 Octave-POETs does not accommodate user definable objective functions, bounds and
42 problem constraints, cooling schedules, different variable types e.g., a mixture of binary
43 and continuous design variables or custom diversity generation routines. Octave-POETs
44 was also not well integrated into a package or source code management (SCM) system.
45 Thus, upgrades to the approach containing new features, or bug fixes were not centrally
46 managed.

47 **Implementation**

48 In this study, we present an open-source implementation of the Pareto optimal ensemble
49 technique in the Julia programming language (JuPOETs). JuPOETs offers many advan-
50 tages and improvements compared to Octave-POETs. JuPOETs takes advantage of the
51 unique features of Julia. Julia is a cross-platform, high-performance programming lan-
52 guage for technical computing that has performance comparable to C but with syntax
53 similar to MATLAB/Octave and Python [25]. Julia also offers a sophisticated compiler,
54 distributed parallel execution, numerical accuracy, and an extensive function library. Fur-
55 ther, the architecture of JuPOETs takes advantage of the first-class function type in Julia
56 allowing user definable behavior for all key aspects of the algorithm, including objective
57 functions, custom diversity generation logic, linear/non-linear parameter constraints (and
58 parameter bounds constraints) as well as custom cooling schedules. Julia’s ability to nat-
59 urally call other languages such as Python or C also allows JuPOETS to be used with
60 models implemented in a variety of languages across many platforms. Additionally, Julia
61 offers a built-in package manager which is directly integrated with GitHub, a popular web-
62 based Git repository hosting service offering distributed revision control and source code
63 management. Thus, JuPOETs can be adapted to many problem types, including mixed
64 binary and continuous variable types, bilevel problems and constrained problems without
65 altering the base algorithm, as was required in the previous POETs implementation.

66 **JuPOETs optimization problem formulation.** JuPOETs solves the \mathcal{K} -dimensional
67 constrained multiobjective optimization problem:

$$\min_{\mathbf{p}} \begin{cases} O_1(\mathbf{x}(t, \mathbf{p}), \mathbf{p}) \\ \vdots \\ O_{\mathcal{K}}(\mathbf{x}(t, \mathbf{p}), \mathbf{p}) \end{cases} \quad (1)$$

68 subject to:

$$\begin{aligned} \mathbf{f}(t, \mathbf{x}(t, \mathbf{p}), \dot{\mathbf{x}}(t, \mathbf{p}), \mathbf{u}(t), \mathbf{p}) &= \mathbf{0} \\ g_1(t, \mathbf{x}(t, \mathbf{p}), \mathbf{u}(t), \mathbf{p}) &\geq 0 \\ &\vdots \\ g_C(t, \mathbf{x}(t, \mathbf{p}), \mathbf{u}(t), \mathbf{p}) &\geq 0 \end{aligned}$$

69 and parameter bound constraints:

$$\mathcal{L} \leq \mathbf{p} \leq \mathcal{U}$$

70 using a modified simulated annealing approach. The quantity t denotes time, $\mathbf{x}(t, \mathbf{p})$
71 denotes the model state (with an initial state \mathbf{x}_0), and $\mathbf{u}(t)$ denotes an input vector. The
72 terms $\mathbf{f}(t, \mathbf{x}(t, \mathbf{p}), \dot{\mathbf{x}}(t, \mathbf{p}), \mathbf{u}(t), \mathbf{p})$ denote the system of model equations (e.g., differential
73 equations, differential algebraic equations or linear/non-linear algebraic equations) where
74 \mathbf{p} denotes the unknown parameter vector ($\mathcal{D} \times 1$). The parameter search can be subject
75 to parameter bound constraints, where \mathcal{L} and \mathcal{U} denote the lower and upper parameter
76 bounds, respectively as well as \mathcal{C} problem specific constraints $g_i(t, \mathbf{x}(t, \mathbf{p}), \mathbf{u}(t), \mathbf{p}), i =$
77 $1, \dots, \mathcal{C}$.

78 JuPOETs integrates simulated annealing with Pareto optimality to estimate parameter
79 sets on or near the optimal tradeoff surface between competing training objectives (Fig.
80 1 and Algorithm 1). The central idea of POETs is a mapping between the value of the
81 objective vector evaluated at \mathbf{p}_{i+1} (parameter guess at iteration $i + 1$) and Pareto rank.
82 JuPOETs calculates the performance of a candidate parameter set \mathbf{p}_{i+1} by calling the
83 user defined `objective` function; `objective` takes a parameter set as an input and returns
84 a $\mathcal{K} \times 1$ objective vector. Candidate parameter sets are generated by the user supplied
85 `neighbor` function. The error vector associated with \mathbf{p}_{i+1} is ranked using the builtin Pareto

86 rank function, by comparing the current error at iteration $i + 1$ to the error archive \mathcal{O}_i (all
87 error vectors up to iteration $i - 1$ meeting a ranking criteria). Pareto rank is a measure
88 of distance from the trade-off surface; parameter sets on or near the optimal trade-off
89 surface between the objectives have a rank equal to 0 (no other current parameter sets
90 are better). Sets with increasing non-zero rank are progressively further away from the
91 optimal trade-off surface. Thus, a parameter set with a rank = 0 is *better* in a trade-off
92 sense than rank > 0 . We implemented the Fonseca and Fleming ranking scheme in the
93 builtin `rank` function [26]:

$$\text{rank}(\mathcal{O}_{i+1}(\mathbf{p}_{i+1}) \mid \mathcal{O}_i) = r \quad (2)$$

94 where rank r is the number of parameter sets that dominate (are better than) parameter
95 set \mathbf{p}_{i+1} , and $\mathcal{O}_{i+1}(\mathbf{p}_{i+1})$ denotes the objective vector evaluated at \mathbf{p}_{i+1} . We used the
96 Pareto rank to inform the SA calculation. The parameter set \mathbf{p}_{i+1} was accepted or rejected
97 by the SA, by calculating an acceptance probability $\mathcal{P}(\mathbf{p}_{i+1})$:

$$\mathcal{P}(\mathbf{p}_{i+1}) \equiv \exp\{-\text{rank}(\mathcal{O}_{i+1}(\mathbf{p}_{i+1}) \mid \mathcal{O}_i) / T\} \quad (3)$$

98 where T is the computational annealing temperature. As $\text{rank}(\mathcal{O}_{i+1}(\mathbf{p}_{i+1}) \mid \mathcal{O}_i) \rightarrow 0$,
99 the acceptance probability moves toward one, ensuring that we explore parameter sets
100 along the Pareto surface. Occasionally, (depending upon T) a parameter set with a high
101 Pareto rank was accepted by the SA allowing a more diverse search of the parameter
102 space. However, as T is reduced, the probability of accepting a high-rank set occur-
103 ring decreases. Parameter sets could also be accepted by the SA but *not* permanently
104 archived in \mathcal{S}_i . Only parameter sets with rank less than or equal to threshold (rank ≤ 4 by
105 default) were included in \mathcal{S}_i , where the archive was re-ranked and filtered after every new
106 parameter set was accepted. Parameter bounds were implemented in the `neighbor` func-
107 tion as box constraints, while problem specific constraints were implemented in objective

108 using a penalty method:

$$O_i + \lambda \sum_{j=1}^c \min\{0, g_j(t, \mathbf{x}(t, \mathbf{p}), \mathbf{u}(t), \mathbf{p})\} \quad i = 1, \dots, \mathcal{K} \quad (4)$$

109 where λ denotes the penalty parameter ($\lambda = 100$ by default). However, because both the
110 neighbor and objective functions are user defined, different constraint implementations
111 are easily defined. JuPOETs can be installed using the Julia package manager from the
112 JuPOETs repository at <https://github.com/varnerlab/POETs.jl>. Sample code is included in
113 the `sample/biochemical` subdirectory of the JuPOETs repository to help users get started
114 using JuPOETs in their projects.

115 **Results and Discussion**

116 JuPOETs identified optimal or nearly optimal solutions significantly faster than Octave-
117 POETs for a suite of multiobjective test problems (Table 1). The wall-clock time for JuPO-
118 ETs and Octave-POETs was measured for 10 independent trials for each of the test prob-
119 lems. The same cooling, neighbor, acceptance, and objective logic was employed
120 between the implementations, and all other parameters were held constant. For each test
121 function, the search domain was partitioned into 10 segments, where an initial parameter
122 guess was drawn from each partition. The number of search steps for each temperate
123 was $\mathcal{I} = 10$ for all cases, and the cooling parameter was $\alpha = 0.9$. On average, JuPOETs
124 identified optimal or near optimal solutions for the suite of test problems six-fold faster (60s
125 versus 400s) than Octave-POETs (Fig. 2). JuPOETs produced the characteristic tradeoff
126 curves for each test problem, given both parameter bound and problem constraints (Fig.
127 3). Thus, JuPOETs estimated an ensemble of solutions to constrained multiobjective op-
128 timization test problems significantly faster than the current Octave implementation. Next,
129 we tested JuPOETs on a proof-of-concept biochemical model identification problem.

130 JuPOETs estimated an ensemble of biochemical models that was consistent with the
131 mean of synthetic training data (Fig. 4). Four synthetic training data sets were gener-
132 ated from a prototypical biochemical network consisting of 6 metabolites and 7 reactions
133 (Fig. 4, inset right). We considered a common case in which the same measurements
134 were made on four hypothetical cell types, each having the same biological connectivity
135 but different performance. Network dynamics were modeled using the hybrid cybernetic
136 model with elementary modes (HCM-EM) approach of Ramkrishna and coworkers [27].
137 In the HCM-EM approach, metabolic networks are first decomposed into a set of ele-
138 mentary modes (EMs) (chemically balanced steady-state pathways, see [28]). Dynamic
139 combinations of elementary modes are then used to characterize network behavior. Each
140 elementary mode is catalyzed by a pseudo enzyme; thus, each mode has both kinetic and

141 enzyme synthesis parameters. The proof of concept network generated 6 EMs, resulting
142 in 13 model parameters. The synthetic data was generated by randomly varying these
143 parameters. JuPOETs produced an ensemble of approximately $\dim \mathcal{S} \simeq 13,000$ parame-
144 ters that captured the mean of the measured data sets for extracellular metabolites and
145 cellmass (Fig. 4A and B). JuPOETs minimized the difference between the simulated and
146 measured values for A_e , B_e , C_e and cellmass, where the residual for each data set was
147 treated as a single objective (leading to four objectives). The 95% confidence estimate
148 produced by the ensemble was consistent with the mean of the measured data, despite
149 having significant uncertainty in the training data. JuPOETs produced a consensus esti-
150 mate of the synthetic data by calculating optimal trade-offs between the training data sets
151 (Fig. 4C). Multiple trade-off fronts were visible in the objective plots, for example between
152 data set 3 (O_3) and data set 2 (O_2). Thus, without a multiobjective approach, it would be
153 challenging to capture these data sets as fitting one leads to decreased performance on
154 the other. However, the ensemble contained parameter sets that described each data set
155 independently (Fig. 5). Thus, JuPOETs produced an ensemble of parameters that gave
156 the mean of the training data for conflicting data sets, while simultaneously estimating
157 parameter sets that performed well on each individual objective function.

158 **Conclusions**

159 JuPOETs is a significant advance over the previous POETs implementation. It offers im-
160 proved performance and is highly adaptable to different problem types. We demonstrated
161 JuPOETs on a suite of test problems, and a proof-of-concept biochemical model. How-
162 ever, there are several areas that could be explored further to improve JuPOETs. First,
163 JuPOETs should be compared with other multiobjective evolutionary algorithms (MOEAs)
164 to determine its relative performance on test and real world problems. Many evolution-
165 ary approaches e.g., the nondominated sorting genetic algorithm (NSGA) family of algo-

166 rithms, have been adapted to solve multiobjective optimization problems [29, 30]. It is
167 unclear if JuPOETs will perform as well as these other approaches; one potential advan-
168 tage that JuPOETs may have is the local refinement step which temporarily reduces the
169 problem to a single objective formulation. Previously, this hybrid approach led to better
170 convergence on a proof-of-concept signal transduction model [23]. For many real world
171 parameter estimation problems, the bulk of the execution time is spent evaluating the
172 objective functions. One strategy to improve performance could be to optimize surro-
173 gates [31], while another would be parallel execution of the objective functions. Currently,
174 JuPOETs serially evaluates the objective function vector. However, parallel evaluation of
175 the objective functions could be easily implemented using a variety of techniques without
176 changing the main run loop of JuPOETs. Because of the flexible function pointer archi-
177 tecture of JuPOETs, the only changes required are in the user defined objective func-
178 tion. Taken together, JuPOETs has demonstrated improved flexibility, and performance
179 over POETs in parameter identification and ensemble generation for multiple objectives.
180 JuPOETs has the potential for widespread use due to the flexibility of the implementation,
181 and the high level syntax and distribution tools native to Julia.

182 **Competing interests**

183 The authors declare that they have no competing interests.

184 **Funding**

185 This study was supported by an award from the National Science Foundation (NSF CBET-
186 0955172) and the National Institutes of Health (NIH HL110328) to J.B, and by a National
187 Science Foundation Graduate Research Fellowship (DGE-1144153) to D.B.

188 **Author's contributions**

189 J.V developed the software presented in this study. M.M and M.V developed the proof-of-
190 concept biochemical model. The manuscript was prepared and edited for publication by
191 D.B, J.B. and J.V.

192 **References**

- 193 1. Gadkar KG, Varner J, Doyle FJ (2005) Model identification of signal transduction
194 networks from data using a state regulator problem. *Syst Biol (Stevenage)* 2: 17–30.
- 195 2. Gennemark P, Wedelin D (2009) Benchmarks for identification of ordinary differential
196 equations from time series data. *Bioinformatics* 25: 780-6.
- 197 3. Battogtokh D, Asch D, Case M, Arnold J, Shüttler H (2002) An ensemble method for
198 identifying regulatory circuits with special reference to the qa gene cluster of *Neu-*
199 *rospora crassa*. *Proc Natl Acad Sci U S A* 99: 16904-16909.
- 200 4. L Kuepfer USJS M Peter (2007) Ensemble modeling for analysis of cell signaling
201 dynamics. *Nat Biotech* 25: 1001-1006.
- 202 5. K S Brown JPS (2003) Statistical mechanical approaches to models with many poorly
203 known parameters. *Phys Rev E* 68: 021904:1-9.
- 204 6. Palmer T, Shutts G, Hagedorn R, Doblas-Reyes F, Jung T, et al. (2005) Representing

- 205 model uncertainty in weather and climate prediction. *Ann Rev Earth and Planetary*
206 *Sci* 33: 163-193.
- 207 7. Gutenkunst RN, Waterfall JJ, Casey FP, Brown KS, Myers CR, et al. (2007) Univer-
208 sally sloppy parameter sensitivities in systems biology models. *PLoS Comput Biol* 3:
209 1871–1878.
- 210 8. Song S, Varner J (2009) Modeling and Analysis of the Molecular Basis of Pain in
211 Sensory Neurons. *PLoS ONE* 4: e6758 - e6772.
- 212 9. Tasseff R, Nayak S, Salim S, Kaushik P, Rizvi N, et al. (2010) Analysis of the molecular
213 networks in androgen dependent and independent prostate cancer revealed fragile
214 and robust subsystems. *PLoS One* 5: e8864.
- 215 10. Tasseff R, Nayak S, Song SO, Yen A, Varner JD (2011) Modeling and analysis
216 of retinoic acid induced differentiation of uncommitted precursor cells. *Integr Biol*
217 *(Camb)* 3: 578-91.
- 218 11. Tran LM, Rizk ML, Liao JC (2008) Ensemble modeling of metabolic networks. *Biophys*
219 *J* 95: 5606-17.
- 220 12. Tan Y, Rivera JGL, Contador CA, Asenjo JA, Liao JC (2011) Reducing the allowable
221 kinetic space by constructing ensemble of dynamic models with the same steady-
222 state flux. *Metab Eng* 13: 60-75.
- 223 13. Contador CA, Rizk ML, Asenjo JA, Liao JC (2009) Ensemble modeling for strain de-
224 velopment of l-lysine-producing escherichia coli. *Metabolic Engineering* 11: 221 -
225 233.
- 226 14. Tan Y, Liao JC (2012) Metabolic ensemble modeling for strain engineers. *Biotechnol*
227 *J* 7: 343-53.
- 228 15. Lee Y, Lafontaine Rivera JG, Liao JC (2014) Ensemble modeling for robustness anal-
229 ysis in engineering non-native metabolic pathways. *Metab Eng* 25: 63-71.
- 230 16. Khodayari A, Zomorodi AR, Liao JC, Maranas CD (2014) A kinetic model of es-

- 231 cherichia coli core metabolism satisfying multiple sets of mutant flux data. *Metab Eng*
232 25: 50-62.
- 233 17. Luan D, Zai M, Varner JD (2007) Computationally derived points of fragility of a human
234 cascade are consistent with current therapeutic strategies. *PLoS Comput Biol* 3:
235 e142.
- 236 18. Song SO, Varner J (2009) Modeling and analysis of the molecular basis of pain in
237 sensory neurons. *PLoS One* 4: e6758.
- 238 19. Nayak S, Siddiqui JK, Varner JD (2011) Modelling and analysis of an ensemble of
239 eukaryotic translation initiation models. *IET Syst Biol* 5: 2.
- 240 20. Song SO, Song SOK, Hogg J, Peng ZY, Parker R, et al. (2012) Ensemble models of
241 neutrophil trafficking in severe sepsis. *PLoS Comput Biol* 8: e1002422.
- 242 21. Luan D, Szlam F, Tanaka KA, Barie PS, Varner JD (2010) Ensembles of uncertain
243 mathematical models can identify network response to therapeutic interventions. *Mol*
244 *Biosyst* 6: 2272-86.
- 245 22. Lequieu J, Chakrabarti A, Nayak S, Varner JD (2011) Computational modeling and
246 analysis of insulin induced eukaryotic translation initiation. *PLoS Comput Biol* 7:
247 e1002263.
- 248 23. Song SO, Chakrabarti A, Varner JD (2010) Ensembles of signal transduction models
249 using pareto optimal ensemble techniques (poets). *Biotechnol J* 5: 768-80.
- 250 24. Eaton JW, Bateman D, Hauberg S (2009) GNU Octave version 3.0.1 manual: a high-
251 level interactive language for numerical computations. North Charleston, SC, USA:
252 CreateSpace Independent Publishing Platform.
- 253 25. Bezanson J, Edelman A, Karpinski S, Shah VB (2015) Julia: A fresh approach to
254 numerical computing .
- 255 26. Fonseca C, Fleming PJ (1993) Genetic Algorithms for Multiobjective Optimization:
256 Formulation, Discussion and Generalization. In: Proceedings of the 5th International

- 257 Conference on Genetic Algorithms. pp. 416 - 423.
- 258 27. Kim J, Varner J, Ramkrishna D (2008) A hybrid model of anaerobic e. coli gjt001:
259 Combination of elementary flux modes and cybernetic variables. Biotechnol Prog 24:
260 993–1006.
- 261 28. Schuster S, Fell DA, Dandekar T (2000) A general definition of metabolic pathways
262 useful for systematic organization and analysis of complex metabolic networks. Nat
263 Biotechnol 18: 326-32.
- 264 29. Kalyanmoy D, Pratap A, Agarwal S, Meyarivan T (2002) A Fast and Elitist Multiobjec-
265 tive Genetic Algorithm: NSGA-II. IEEE Trans Evol Comp 6: 182 - 197.
- 266 30. Huband S, Hingston P, Barone L, While L (2006) A Review of Multiobjective Test
267 Problems and a Scalable Test Problem Toolkit. IEEE Trans Evol Comp 10: 477 -
268 506.
- 269 31. Booker A, Dennis J, Frank P, Serafini D, Torczon V, et al. (1999) A rigorous framework
270 for optimization of expensive functions by surrogates. Struct Optim 17: 1 - 13.

input : User specified neighbor, objective, acceptance and cooling functions. Initial parameter guess ($\mathcal{D} \times 1$)

Output: Rank archive \mathcal{R} , parameter solution archive \mathcal{S} and objective archive \mathcal{O}

```
1 initialize:  $\mathcal{R}$ ,  $\mathcal{S}$  and  $\mathcal{O}$  using initial guess;
2 initialize:  $T \leftarrow 1.0$ ;
3 initialize:  $T_{min} \leftarrow 1/10000$ ;
4 initialize: Maximum number of steps per temperature  $\mathcal{I}$ ;

5 while  $T > T_{min}$  do
6      $i \leftarrow 1$ ;
7     while  $i < \mathcal{I}$  do
8         // Generate a new parameter solution using user neighbor function
9          $\mathbf{p}_{i+1} \leftarrow \text{user-function}::\text{neighbor}(\mathbf{p}^*)$ ;
10
11        // Evaluate  $\mathbf{p}_{i+1}$  using user objective function
12         $\mathbf{o}_{i+1} \leftarrow \text{user-function}::\text{objective}(\mathbf{p}_{i+1})$ ;
13
14        Add  $\mathbf{p}_{i+1}$  to solution archive  $\mathcal{S}$ ;
15        Add  $\mathbf{o}_{i+1}$  to objective archive  $\mathcal{O}$ ;
16
17        // Calculate Pareto rank of solutions in  $\mathcal{O}$  using builtin rank function
18         $\mathcal{R} \leftarrow \text{builtin-function}::\text{rank}(\mathcal{O})$ ;
19
20        // Accept  $\mathbf{p}_{i+1}$  into the archive with user defined probability
21         $\mathcal{P} \leftarrow \text{user-function}::\text{acceptance}(\mathcal{R}, T)$ ;
22        if  $\mathcal{P} > \text{rand}$  then
23            // Update the best solution with  $\mathbf{p}_{i+1}$ 
24             $\mathbf{p}^* \leftarrow \mathbf{p}_{i+1}$ ;
25            prune  $\mathcal{S}$ ,  $\mathcal{R}$  and  $\mathcal{O}$  of all solutions above a rank threshold;
26        else
27            Remove  $\mathbf{p}_{i+1}$  from solution archive  $\mathcal{S}$ ;
28            Remove  $\mathbf{o}_{i+1}$  from error archive  $\mathcal{O}$ ;
29        end
30         $i \leftarrow i + 1$ ;
31    end
32
33    // Update  $T$  using the user cooling function
34     $T \leftarrow \text{user-function}::\text{cooling}(T)$ ;
35 end
```

Algorithm 1: Pseudo-code for the main run-loop of JuPOETs. The user specifies the neighbor, acceptance, cooling and objective functions along with an initial parameter guess. The rank archive \mathcal{R} , solution archive \mathcal{S} and objective archive \mathcal{O} are initialized from the initial guess. The initial guess is perturbed in the neighbor function, which generates a new solution whose performance is evaluated using the user supplied objective function. The new solution and objective values are then added to the respective archives and ranked using the builtin rank function. If the new solution is accepted (based upon a probability calculated with the user supplied acceptance function) it is added to the solution and objective archive. This solution is then perturbed during the next iteration of the algorithm. However, if the solution is not accepted, it is removed from the archive and discarded. The computational temperature is adjusted using the user supplied cooling function after each \mathcal{I} iterations.

271 **Figures**

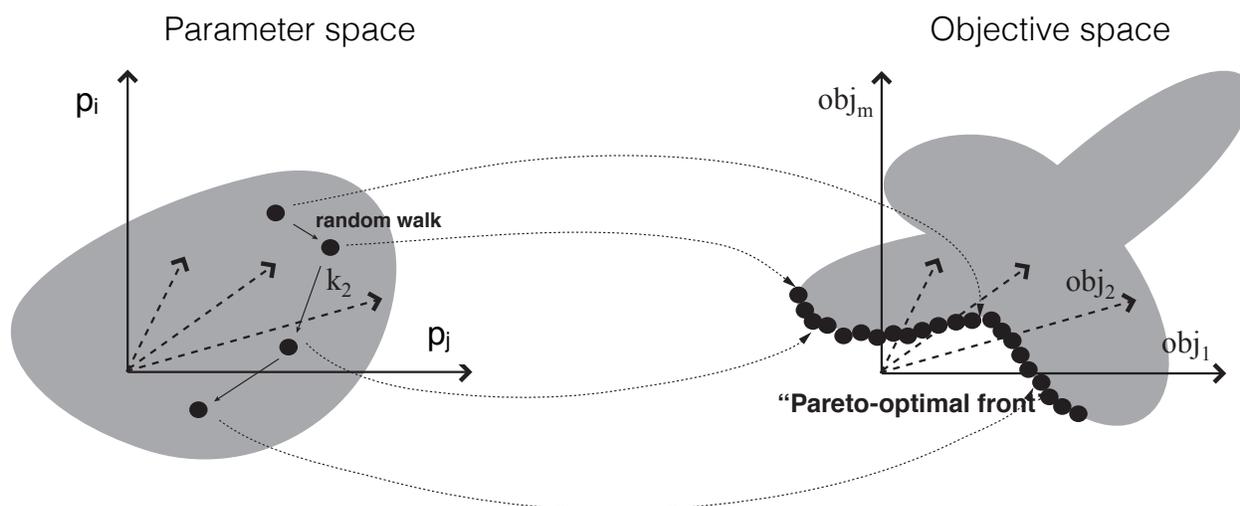


Fig. 1: Schematic of multiobjective parameter mapping. The performance of any given parameter set is mapped into an objective space using a ranking function which quantifies the quality of the parameters. The distance away from the optimal tradeoff surface is quantified using the Pareto ranking scheme of Fonseca and Fleming in JuPOETs.

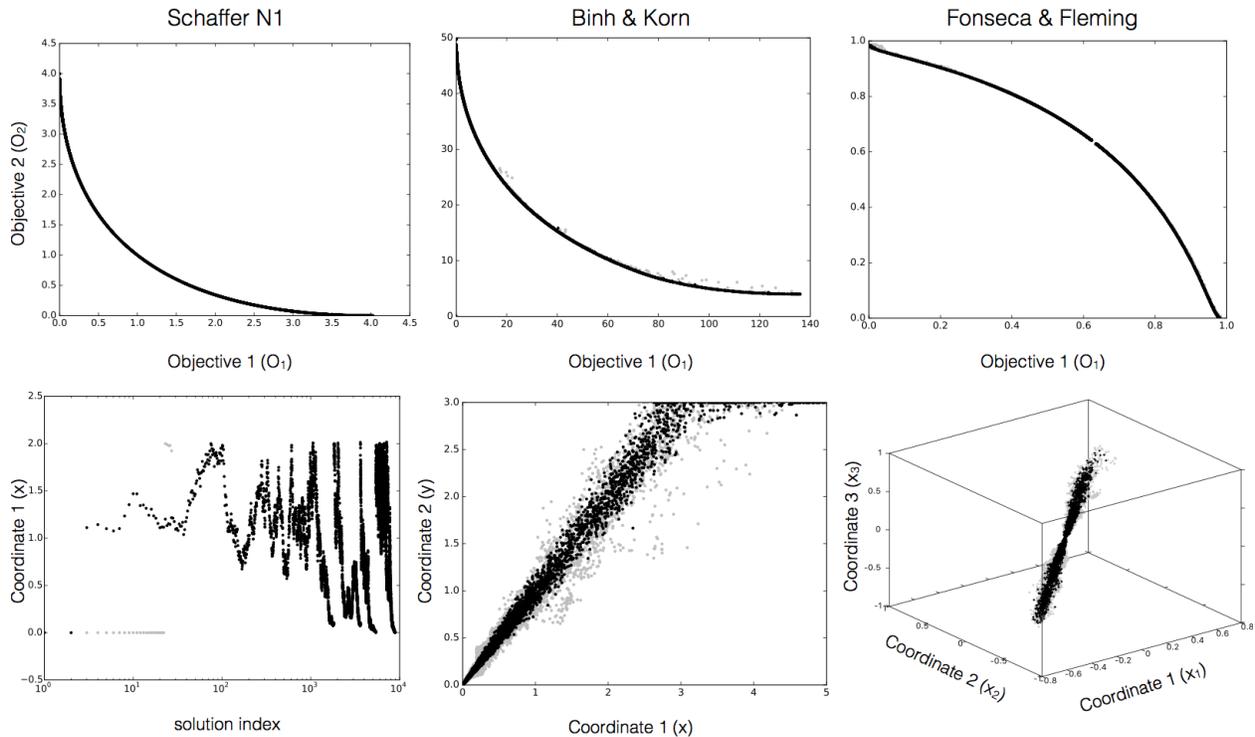


Fig. 2: The performance of JuPOETs on the multi-objective test suite. The execution time (wall-clock) for JuPOETs and POETs implemented in Octave was measured for 10 independent trials for the suite of test problems. The number of steps per temperature $\mathcal{I} = 10$, and the cooling parameter $\alpha = 0.9$ for all cases. The problem domain was partitioned into 10 equal segments, an initial guess was drawn from each segment. For each of the test functions, JuPOETs estimated solutions on (rank zero solutions, black) or near (gray) the optimal tradeoff surface, subject to bounds and problem constraints.

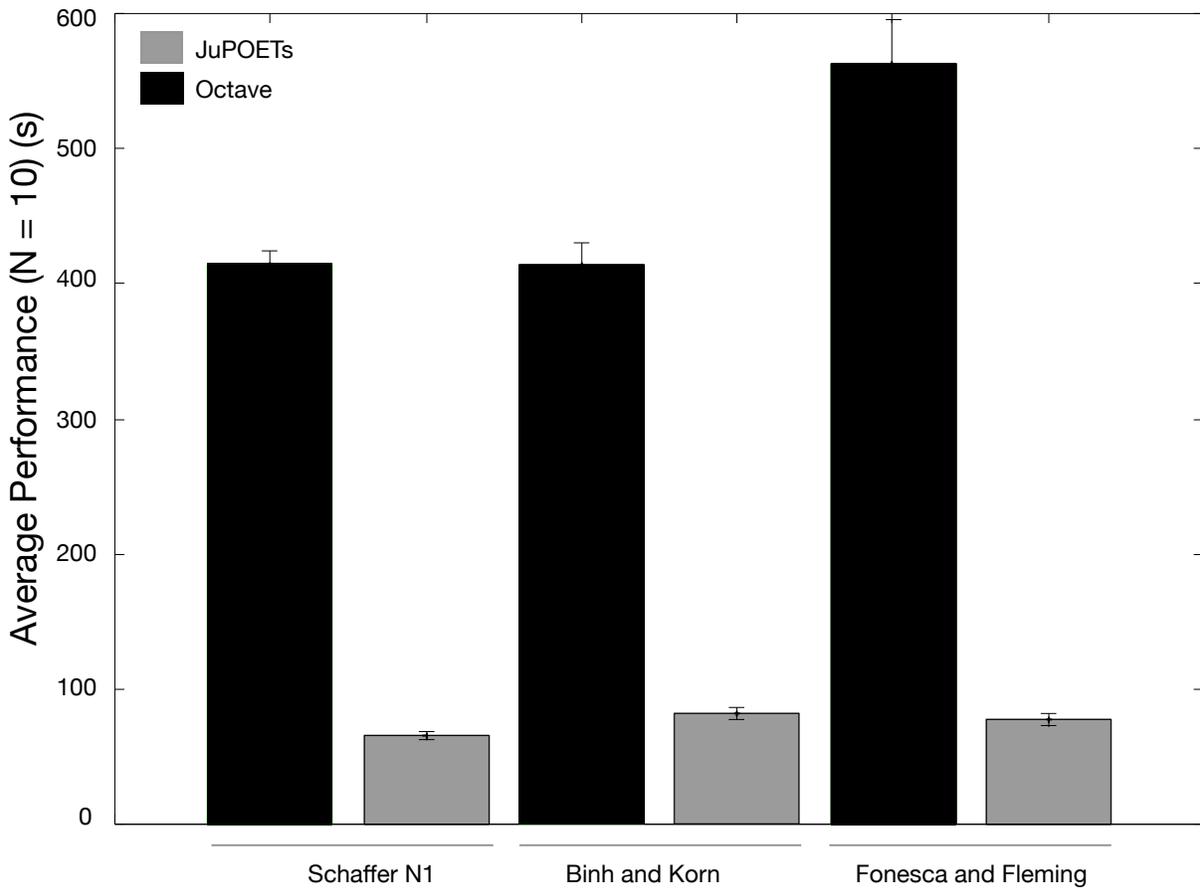


Fig. 3: Representative JuPOETs solutions for problems in the multi-objective test suite. The number of steps per temperature $\mathcal{I} = 10$, and the cooling parameter $\alpha = 0.9$ for all cases. The problem domain was partitioned into 10 equal segments, an initial guess was drawn from each segment. For each of the test functions, JuPOETs estimated solutions on (rank zero solutions, black) or near (gray) the optimal tradeoff surface, subject to bounds and problem constraints.

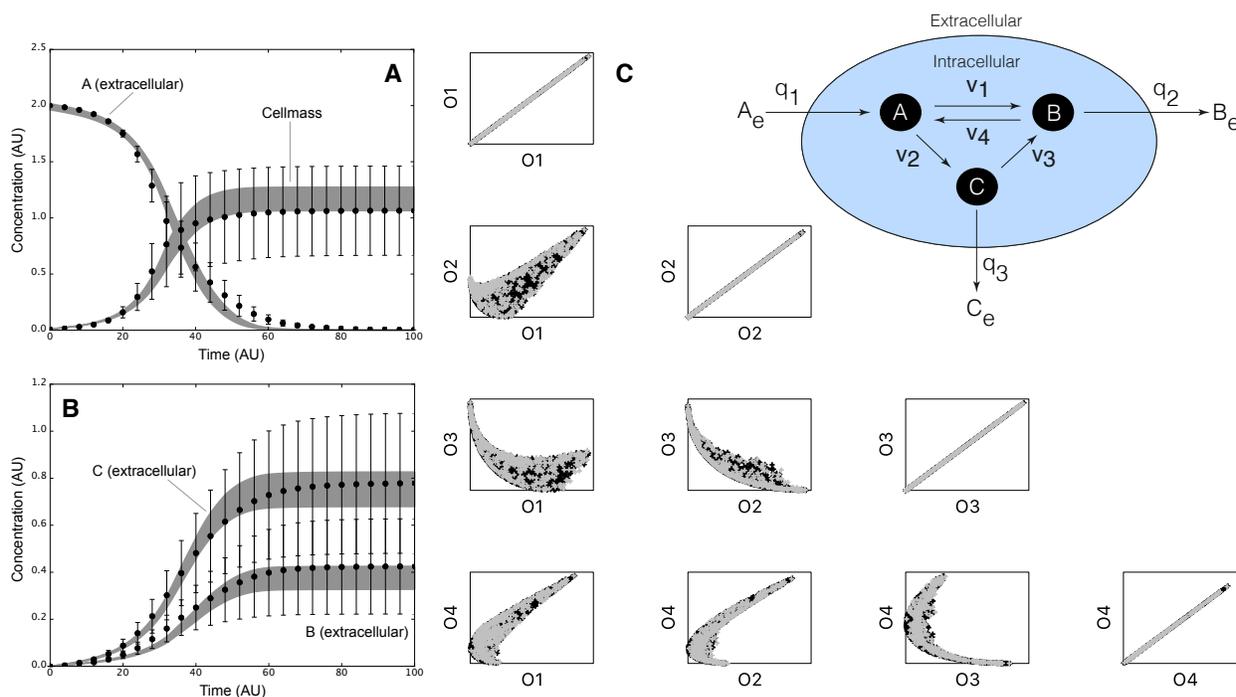


Fig. 4: Proof of concept biochemical network study. Inset right: Prototypical biochemical network with six metabolites and seven reactions modeled using the hybrid cybernetic approach (HCM). Intracellular cellmass precursors A , B , and C are balanced (no accumulation) while the extracellular metabolites A_e , B_e , and C_e are dynamic. The oval denotes the cell boundary, q_j is the j th flux across the boundary, and v_k denotes the k th intracellular flux. Four data sets (each with A_e , B_e , C_e and cellmass measurements) were generated by varying the kinetic constants for each biochemical mode. Each data set was a single objective in the JuPOETs procedure. **A:** Ensemble simulation of extracellular substrate A_e and cellmass versus time. **B:** Ensemble simulation of extracellular substrate B_e and C_e versus time. The gray region denotes the 95% confidence estimate of the mean ensemble simulation. The data points denote mean synthetic measurements, while the error bars denote the 95% confidence estimate of the measurement computed over the four training data sets. **C:** Trade-off plots between the four training objectives. The quantity O_j denotes the j th training objective. Each point represents a member of the parameter ensemble, where black denotes a rank 0 set, while gray denotes rank 1 set.

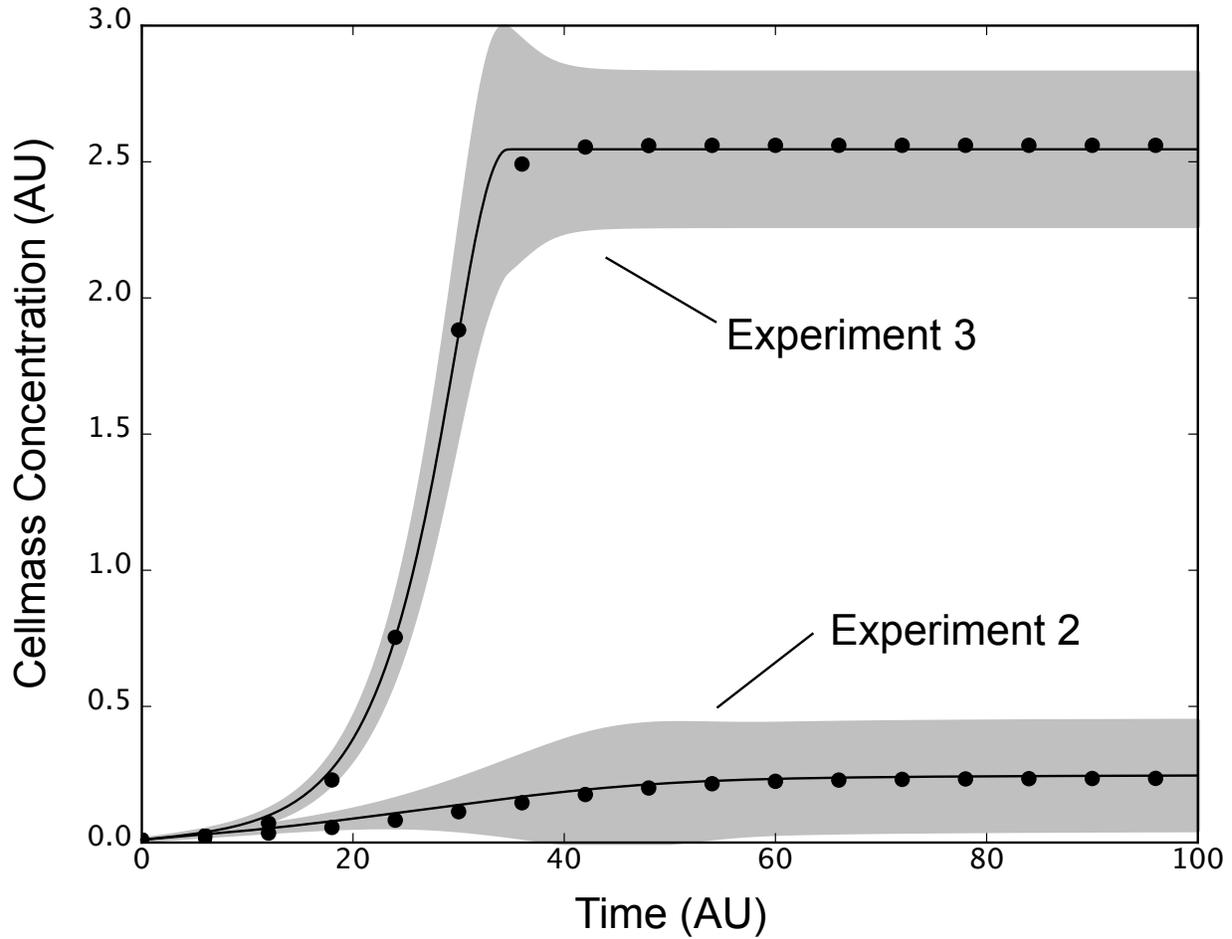


Fig. 5: Experiment to experiment variation is captured by a single ensemble. Cellmass measurements (points) versus time for experiment 2 and 3 were compared with ensemble simulations. The full ensemble was sorted by simultaneously selecting the top 25% of solutions for each objective with rank ≤ 1 . The best fit solution for each objective (line) ± 1 -standard deviation (gray region) for experiment 2 and 3 brackets the training data despite significant differences the training values between the two data sets.

Name	Dimension	Function	Domain	Constraints
Schaffer function	1	$O_1(x) = x^2$ $O_2(x) = (x - 2)^2$	$-10 \leq x \leq 10$	
Binh and Korn function	2	$O_1(x, y) = 4x^2 + 4y^2$ $O_2(x, y) = (x - 5)^2 + (y - 5)^2$	$0 \leq x \leq 5$ $0 \leq y \leq 3$	$g_1(x, y) = (x - 5)^2 + y^2 \leq 25$ $g_2(x, y) = (x - 8)^2 + (y + 3)^2 \geq 7.7$
Fonseca and Fleming function	3	$O_1(x_i) = 1 - \exp\left(-\sum_{i=1}^N \left(x_i - \frac{1}{\sqrt{N}}\right)^2\right)$ $O_2(x_i) = 1 - \exp\left(-\sum_{i=1}^N \left(x_i + \frac{1}{\sqrt{N}}\right)^2\right)$	$-4 \leq x_i \leq 4$	

Table 1: Multi-objective optimization test problems. We tested the JuPOETs implementation on three two-dimensional test problems, with one-, two- and three-dimensional parameter vectors. Each problem had parameter bounds constraints, however, on the Binh and Korn function had additional non-linear problem constraints. For the Fonesca and Fleming problem, $N = 3$.