



MXMC: Generalized Multi-Model Monte Carlo Simulation for Uncertainty Propagation

G.F. Bomarito, J.E. Warner, P.E. Leser, W.P. Leser, L. Morrill

National Aeronautics and Space Administration,
Langley Research Center, Hampton, VA

Multifidelity Uncertainty Propagation

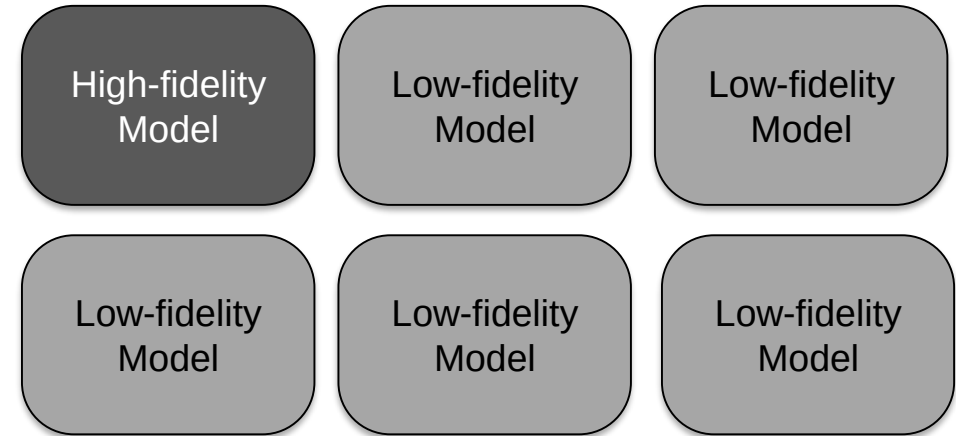


Given:

- Quantity of interest (QOI)
- A high-fidelity model
- M low-fidelity approximations

Find:

- Statistics of the high-fidelity model



- [1] Giles, M B. Operations Research (2008)
- [2] Peherstorfer, B, et al. SIAM Journal on Scientific Computing (2016)
- [3] Gorodetsky, A A., et al. Journal of Computational Physics (2020)
- [4] Bomarito, GF., et al. arXiv:2012.02750 (2020)
- [5] Schaden, D, and E. Ullmann. SIAM/ASA Journal on Uncertainty Quantification (2020):

Multifidelity Uncertainty Propagation

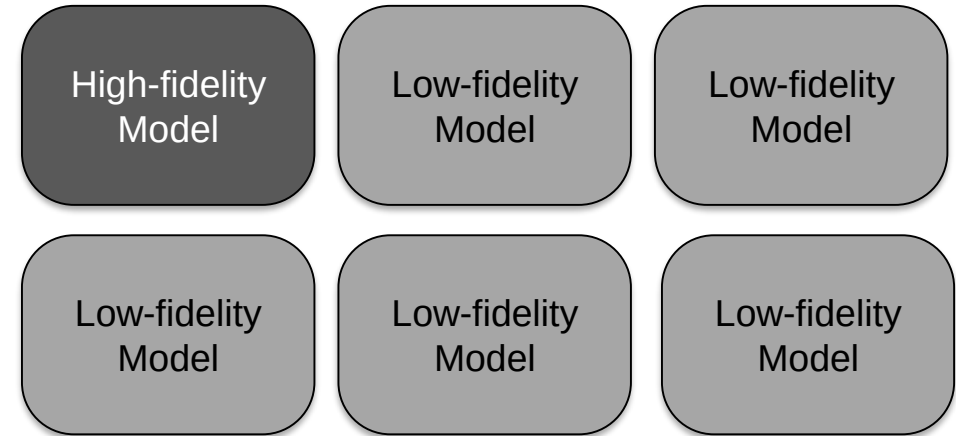


Given:

- Quantity of interest (QOI)
- A high-fidelity model
- M low-fidelity approximations

Find:

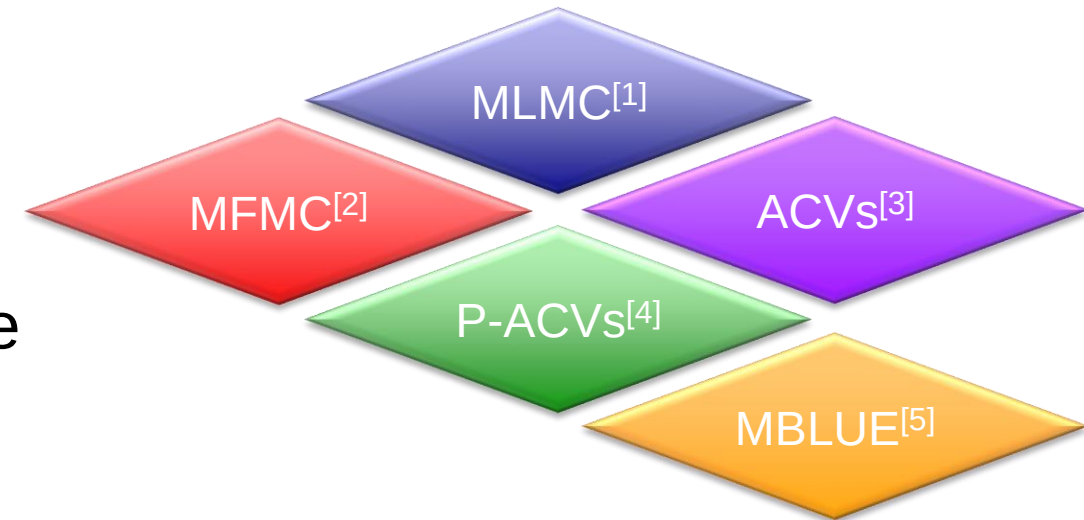
- Statistics of the high-fidelity model



- Many algorithms exist
- Different algorithms for different scenarios

Our Solution:

Make the different algorithms easy to compare



[1] Multi-level Monte Carlo: Giles, M B.

[2] Multifidelity Monte Carlo: Peherstorfer, B, et al.

[3] Approximate Control Variates: Gorodetsky, A A., et al.

[4] Parametrically-defined ACVs: Bomarito, GF., et al.

[5] Multifidelity Best Linear Unbiased Estimators: Schaden, D, and E. Ullmann.

Our Solution: MXMCPy



github.com/nasa/MXMCPy

- Easy to compare algorithms
- Easy to add new algorithms

Minimum working example on github.com/nasa/MXMCPy

```
import numpy as np
from mxmc import Optimizer

model_costs = np.array([1.0, 0.05, 0.001])
covariance_matrix = np.array([[11.531, 11.523, 12.304],
                              [11.523, 11.518, 12.350],
                              [12.304, 12.350, 14.333]])

optimizer = Optimizer(model_costs, covariance_matrix)
opt_result = optimizer.optimize(algorithm="mfmc", target_cost=1000)

print("Optimal variance: ", opt_result.variance)
print("# samples per model: ", opt_result.allocation.get_number_of_samples_per_model())
```

```
Optimal variance:  0.0001423541366151887
# samples per model:  [  145 11649 272410]
```

Our Solution: MXMCPy



github.com/nasa/MXMCPy

- Easy to compare algorithms
- Easy to add new algorithms

Minimum working example on github.com/nasa/MXMCPy

```
import numpy as np
from mxmc import Optimizer

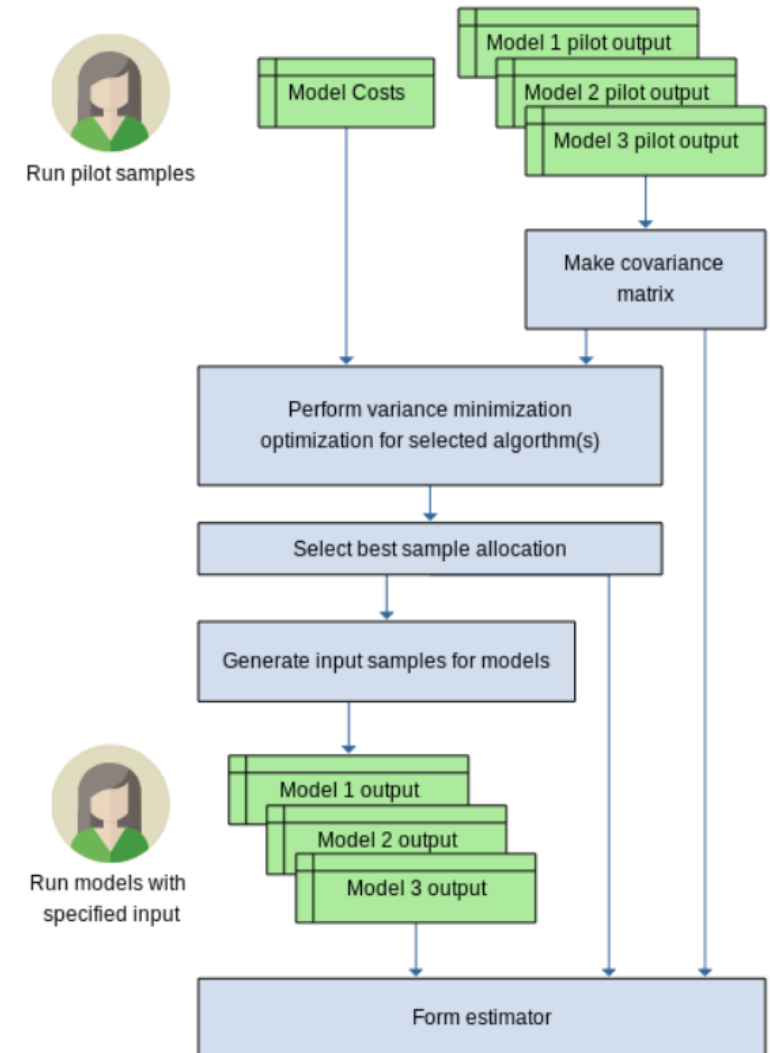
model_costs = np.array([1.0, 0.05, 0.001])
covariance_matrix = np.array([[11.531, 11.523, 12.304],
                              [11.523, 11.518, 12.350],
                              [12.304, 12.350, 14.333]])

optimizer = Optimizer(model_costs, covariance_matrix)
opt_result = optimizer.optimize(algorithm="mfmc", target_cost=1000)

print("Optimal variance: ", opt_result.variance)
print("# samples per model: ", opt_result.allocation.get_number_of_samples_per_model())
```

```
Optimal variance: 0.0001423541366151887
# samples per model: [ 145 11649 272410]
```

Typical MXMCPy Workflow



A study using MXMCPy



Explore how the model scenario
(correlation and costs) affects

1. Optimal sampling strategy lowest variance

Sampling Strategy		Examples
GMF	Generalized multifidelity	MFMC, ACVMF ^[1]
GRD	Generalized recursive difference	MLMC
GIS	Generalized independent samples	ACVIS ^[2]

[1] ACV Multifidelity: Gorodetsky, A A., et al.

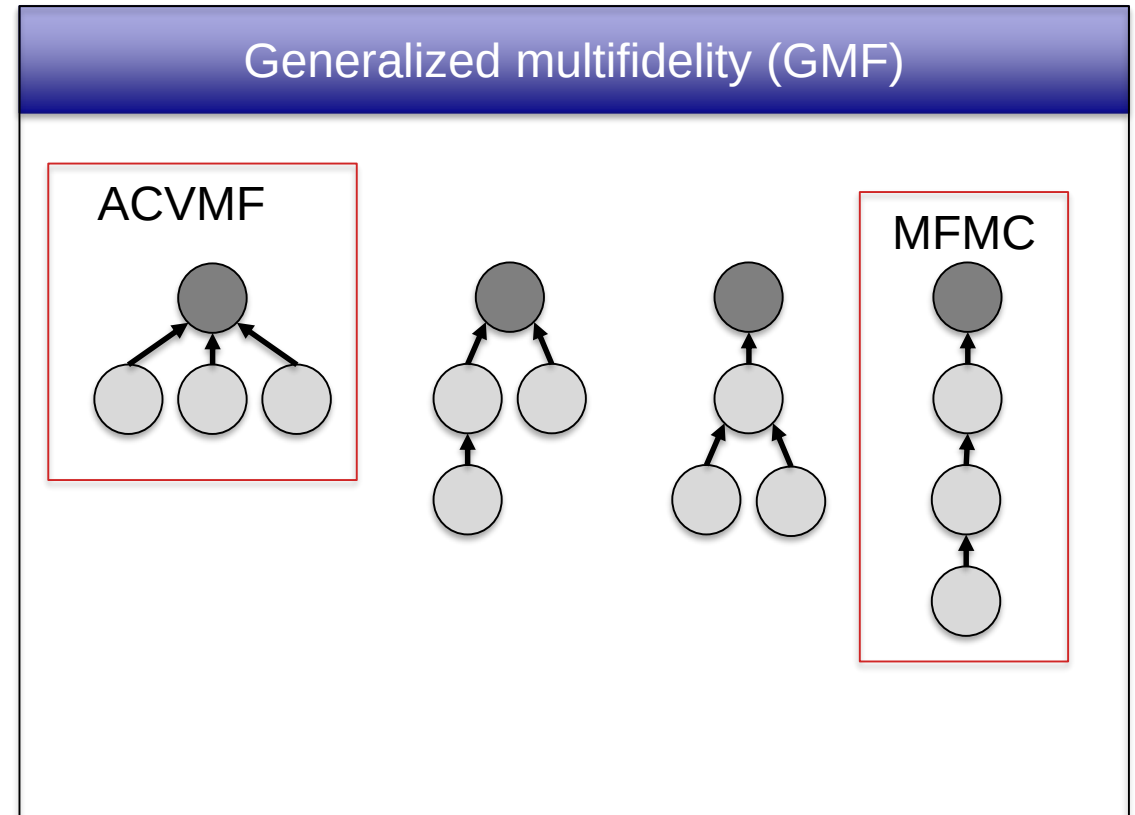
[2] ACV Independent Samples: Gorodetsky, A A., et al

A study using MXMCPy



Explore how the model scenario
(correlation and costs) affects

1. Optimal sampling strategy
lowest variance
2. **Optimal model hierarchy**

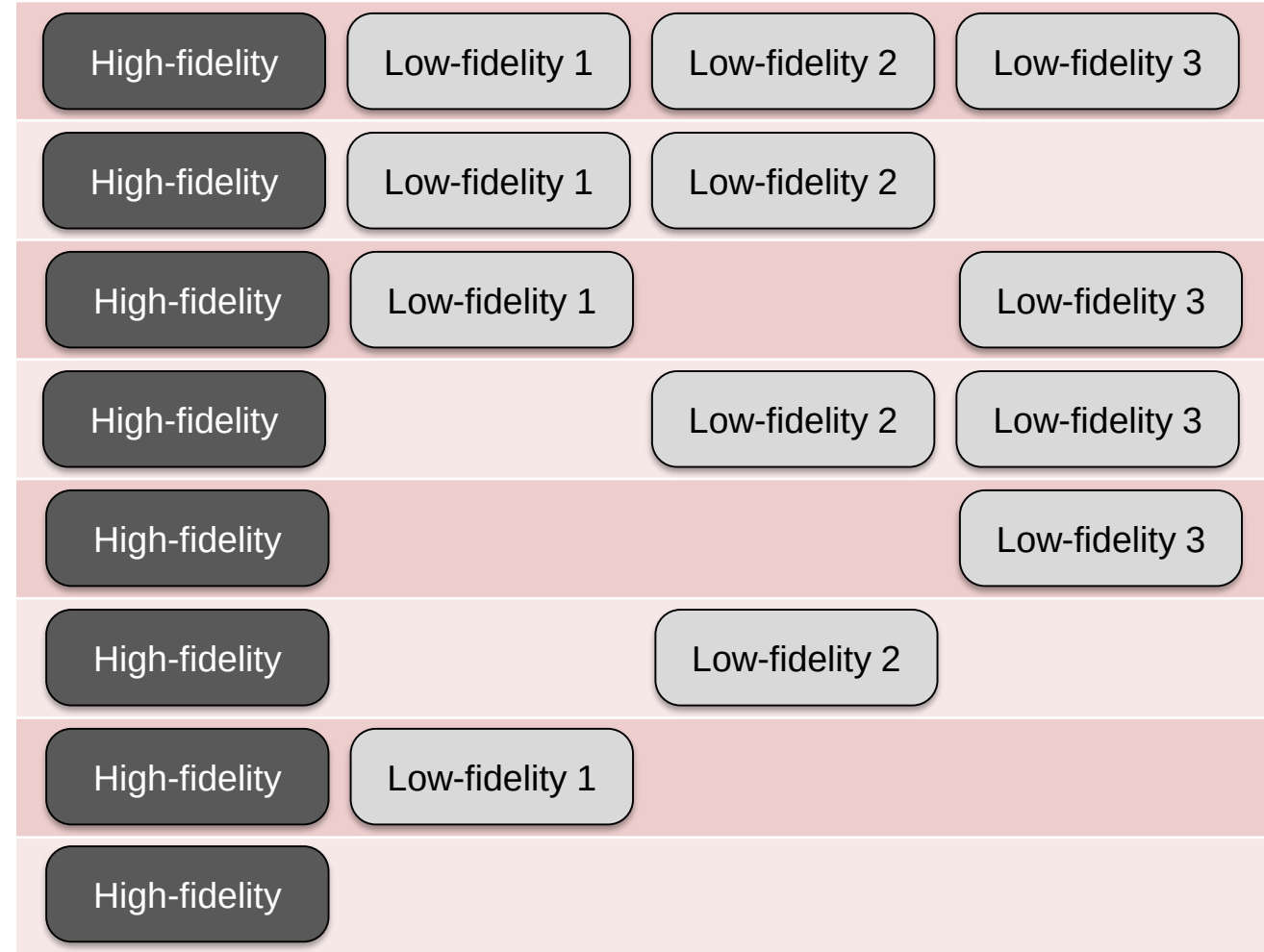


A study using MXMCPy



Explore how the model scenario (correlation and costs) affects

1. Optimal sampling strategy
lowest variance
2. Optimal model hierarchy
3. **Optimal model subset**



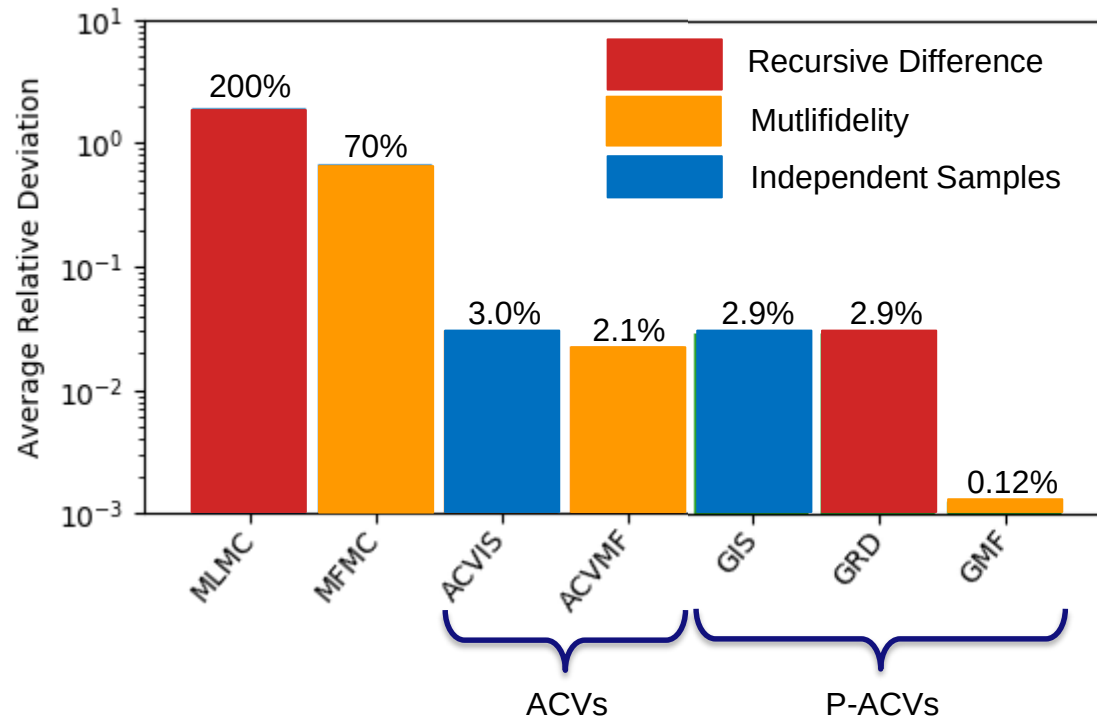
Random Model Scenarios



- ~5 Million
1. Choose number of models [2-6]
 2. Sample random correlation matrix
 3. Sample random variances
 4. Sample random model costs (relative to hifi cost)
 5. Perform allocation optimization for all algorithms (fixed cost budget)

Random Model Scenarios

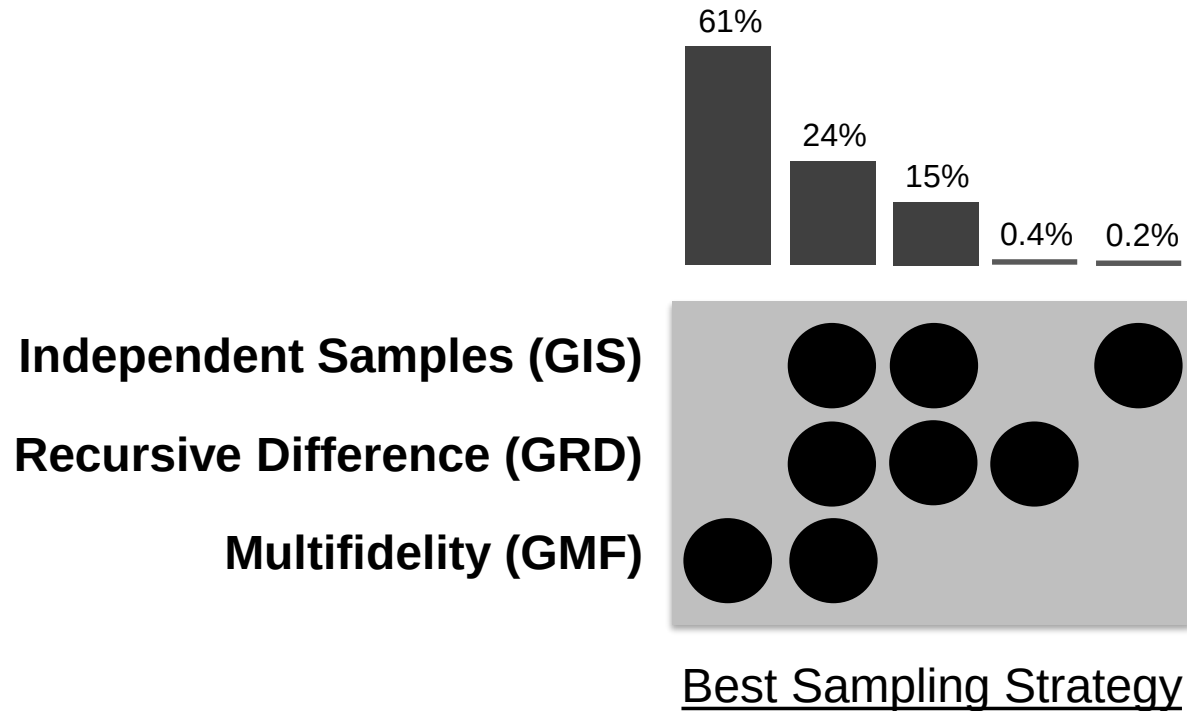
- ~5 Million
1. Choose number of models [2-6]
 2. Sample random correlation matrix
 3. Sample random variances
 4. Sample random model costs (relative to hifi cost)
 5. Perform allocation optimization for all algorithms (fixed cost budget)



Disclaimer:

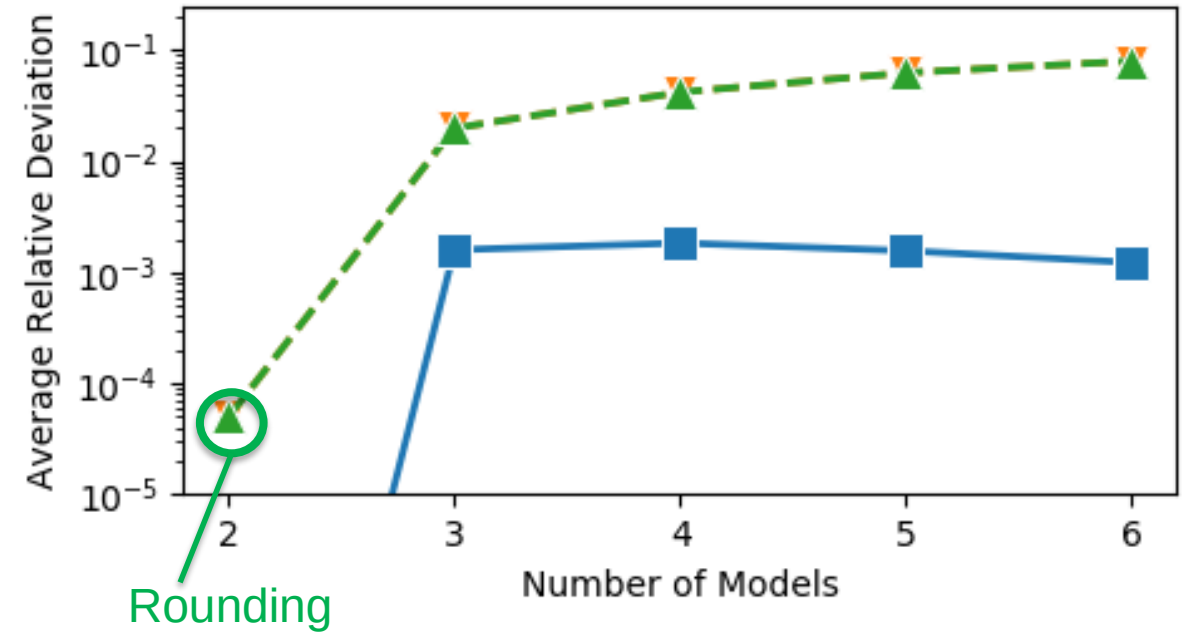
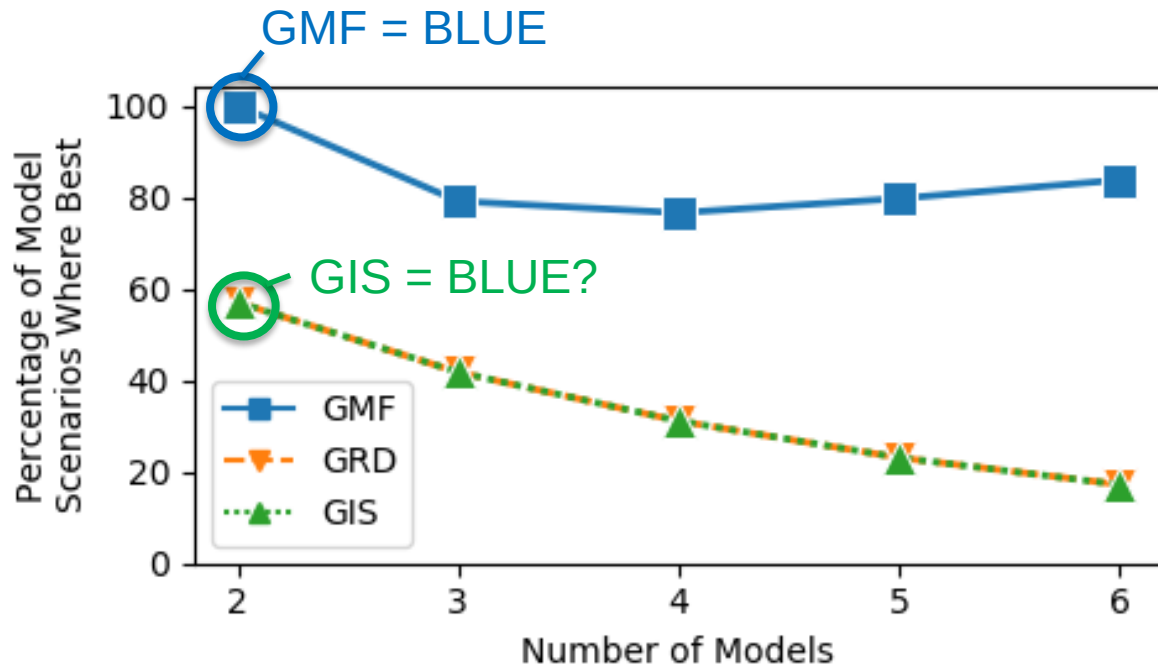
This distribution of model scenarios may not be representative of the distribution of model scenarios encountered in real applications.

Comparison of Sampling Strategies



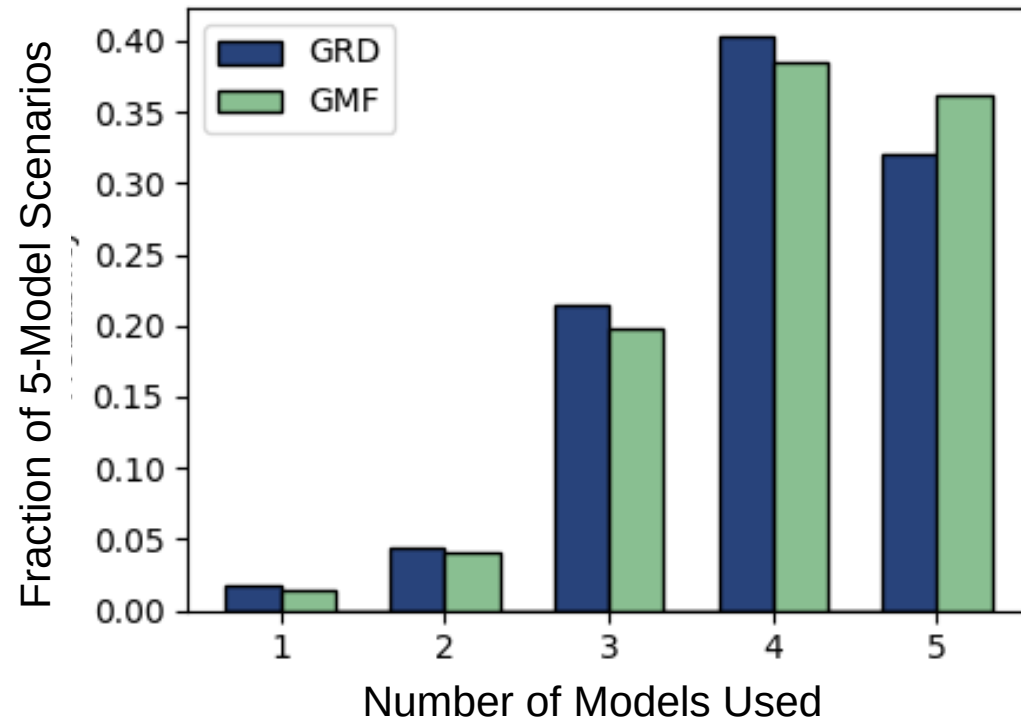
**Multifidelity sampling best
in 85% of model scenarios**

Scalability of Sampling Strategies



GMF scales better to more models

Why Does GMF Scale Better?

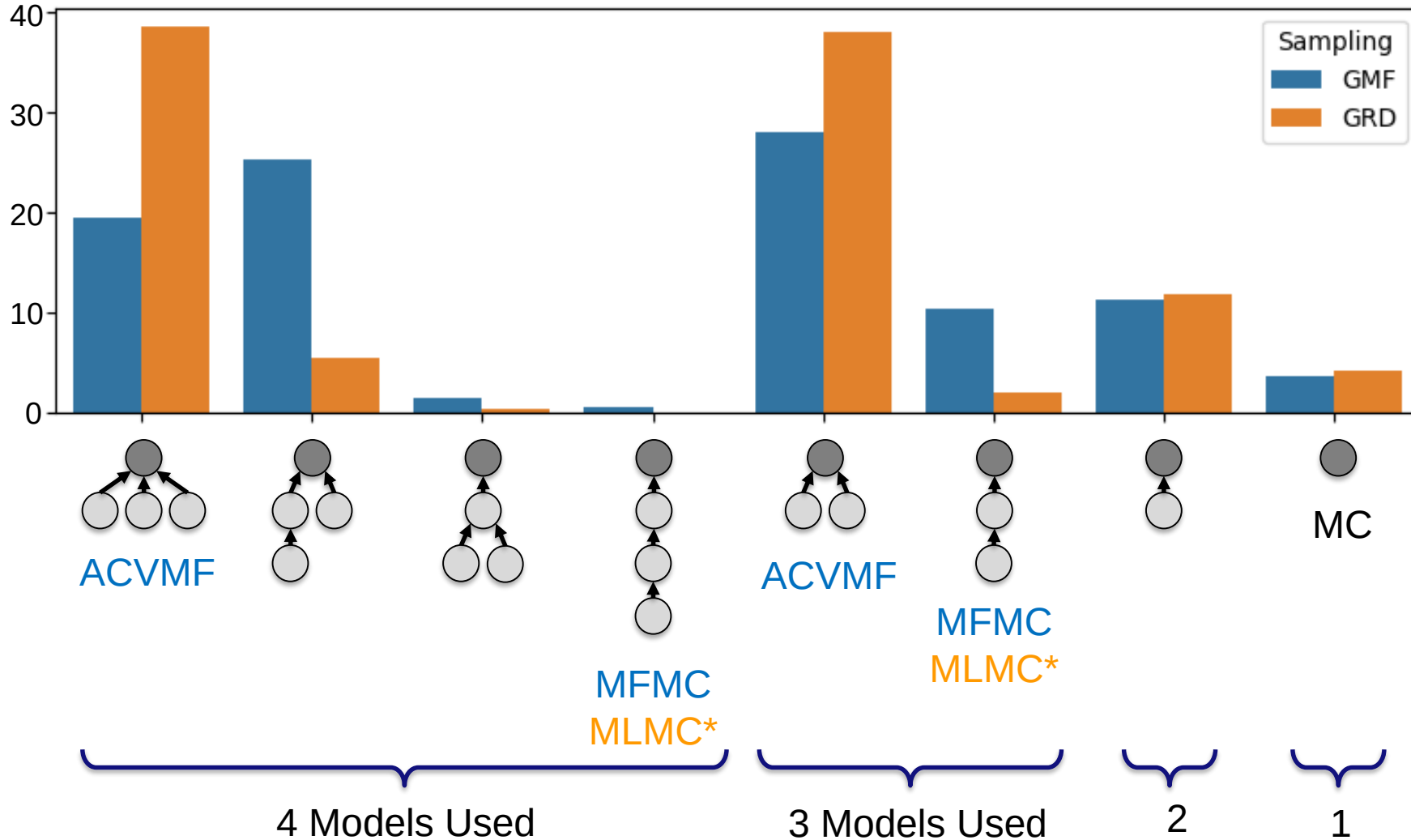


- GMF more frequently utilizes more models
- GMF better than GRD in ~70% of 5-model scenarios
 - GMF must better leverage models even when same models are chosen

Relation to Optimal Model Hierarchy



Percentage of 4-Model Scenarios



GRD:

- 77% flat hierarchies
- 7% deep hierarchies
- 16% 1 or 2 Models

GMF:

- 47% flat hierarchies
- 38% deep hierarchies
- 15% 1 or 2 Models

Sharing samples with the high-fidelity model is important for variance reduction!

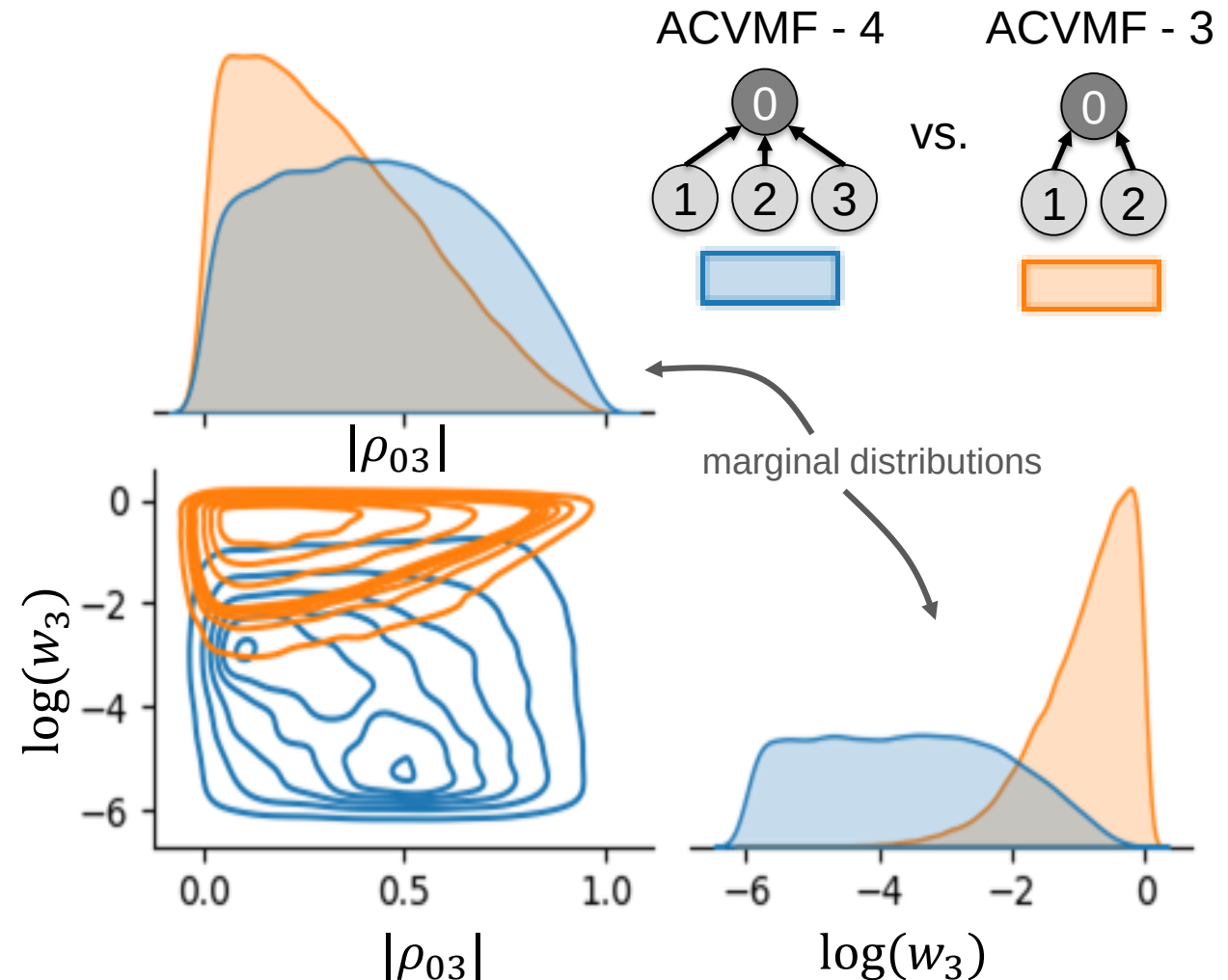
Comparison of Optimal Model Hierarchies



GMF 4-Model Scenarios

- ACVMF - 4 vs. ACVMF - 3
- Most important parameters:
 - Cost w_3
 - Correlation with hifi ρ_{03}
- GRD is similar

Model cost is most important in model selection but can be offset by high correlation.



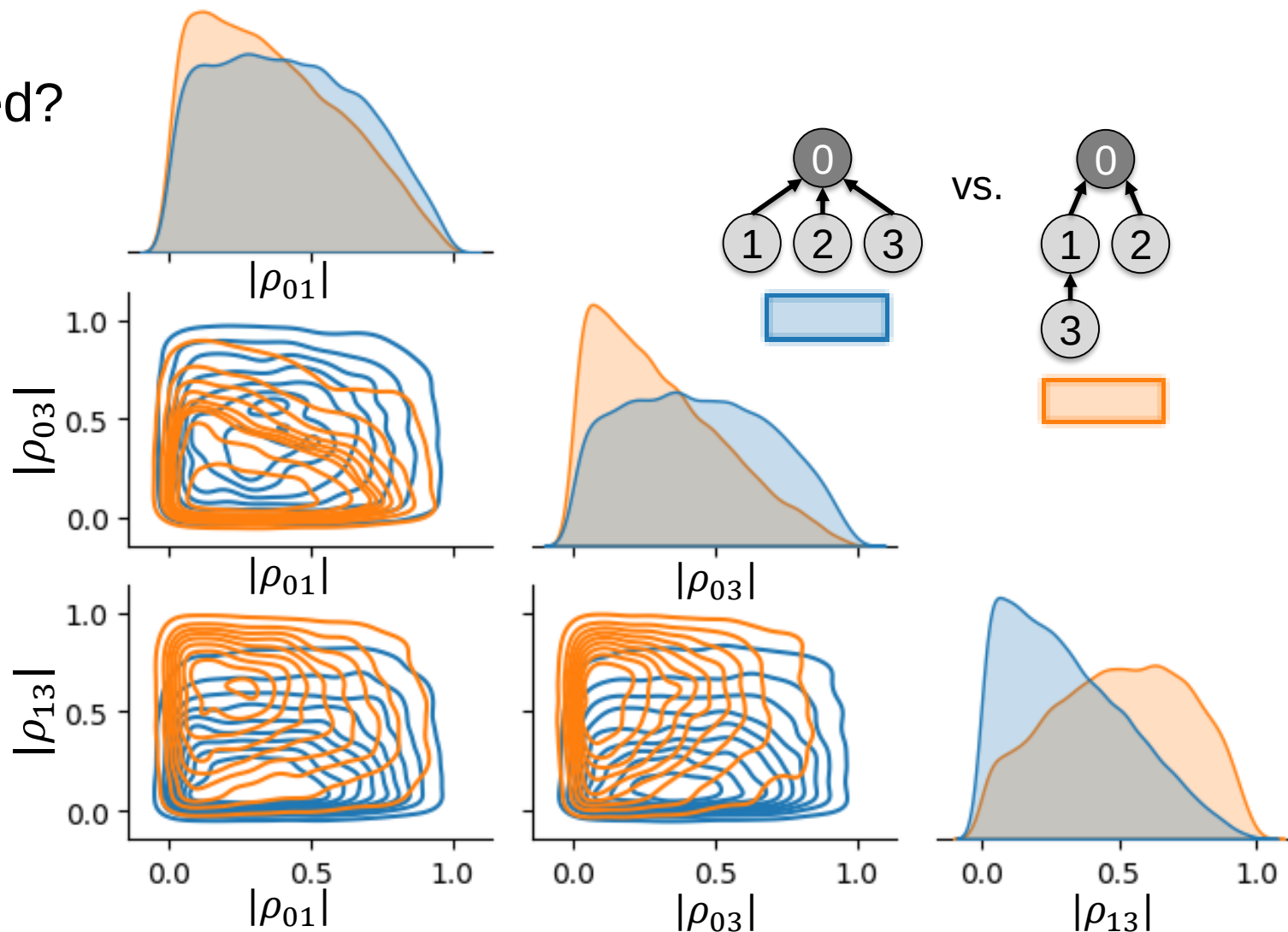
Comparison of Optimal Model Hierarchies



GMF 4-Model Scenarios

- When should depth be added?
- Most important parameters:
 - Correlations between 0,1,3

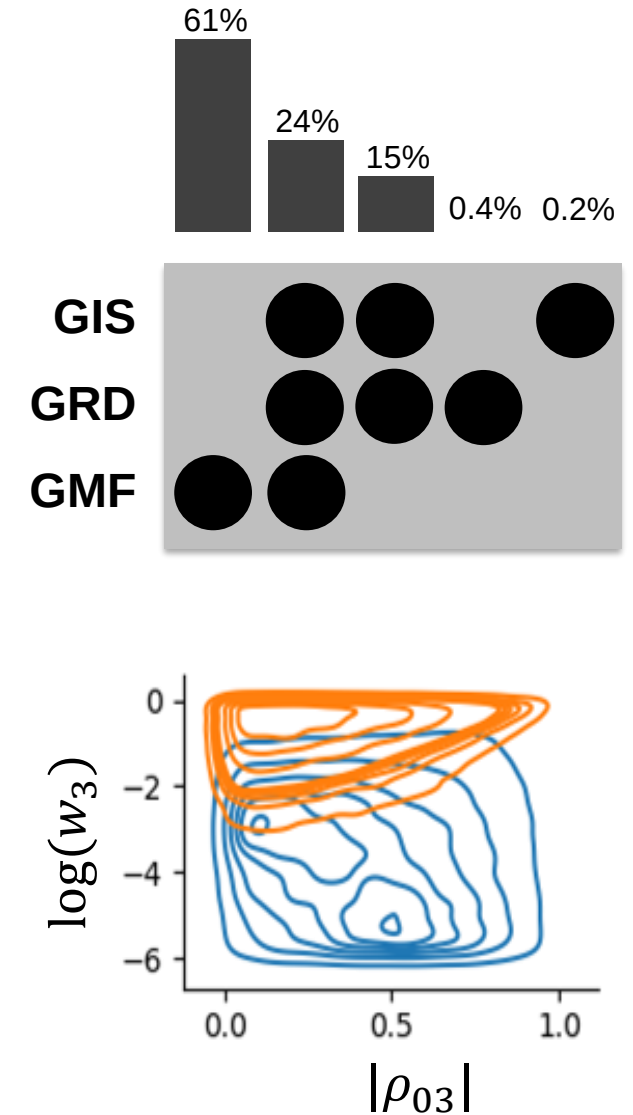
Link to most correlated model
(but not always)



Study Conclusions



- Multifidelity Sampling
 - Usually the best sampling strategy
 - Scales better to more models
 - Likely because lowfi models share more samples with hifi model
- Model cost is most important in model selection but can be offset by high correlation.
- Optimal hierarchies tend to couple models based on highest individual correlations (but not always)



- Enabled systematic study of many multifidelity MC algorithms
- Relatively painless addition of other algorithms
 - MBLUE on the way

Resources:

- github.com/nasa/MXMCPy
- Bomarito, GF., et al. “Multi model monte carlo with python (mxmcpy)” NASA/TM-2020-220585 (2020)
- Bomarito, GF., et al. “On the Optimization of Approximate Control Variates with Parametrically Defined Estimators” arXiv:2012.02750 (2020)