

Correctness and Performance Charts

This version of the document is dated 2022-11-07.

The following charts show the correctness of many of the algorithms in "**Bernoulli Factory Algorithms**" and show their performance in terms of the number of bits they use on average. For each algorithm, and for each of 100 λ values evenly spaced from 0.0001 to 0.9999:

- 500 runs of the algorithm were done. Then...
- The number of bits used by the runs were averaged, as were the return values of the runs (since the return value is either 0 or 1, the mean return value will be in the interval $[0, 1]$). The number of bits used included the number of bits used to produce each coin flip, assuming the coin flip procedure for λ was generated using the `Bernoulli#coin()` method in *bernoulli.py*, which produces that probability in an optimal or near-optimal way.

For each algorithm, if a single run was detected to use more than 5000 bits for a given λ , the entire data point for that λ was suppressed in the charts below.

In addition, for each algorithm, a chart appears showing the minimum number of input coin flips that any fast Bernoulli factory algorithm will need on average to simulate the given function, based on work by Mendo (2019)[¹]. Note that some functions require a growing number of coin flips as λ approaches 0 or 1. Note that for the 2014, 2016, and 2019 algorithms—

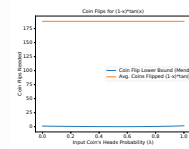
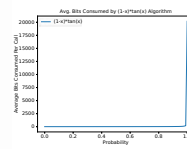
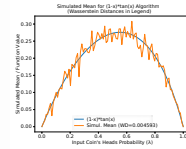
- an ϵ of $1 - (x + c) * 1.001$ was used (or 0.0001 if ϵ would be greater than 1), and
- an ϵ of $(x - c) * 0.9995$ for the subtraction variants.

Points with invalid ϵ values were suppressed. For the low-mean algorithm, an m of $\max(0.49999, x*c*1.02)$ was used unless noted otherwise.

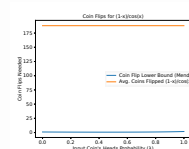
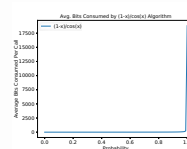
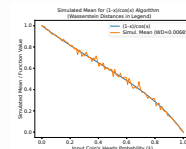
0.1 The Charts

| Algorithm | Simulated Mean | Average Bits Consumed | Coin Flips |
|-----------|----------------|-----------------------|------------|
|-----------|----------------|-----------------------|------------|

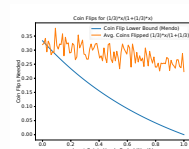
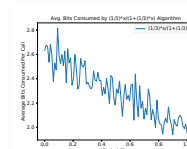
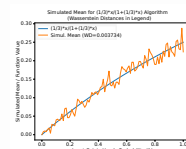
$$(1-x)*\tan(x)$$



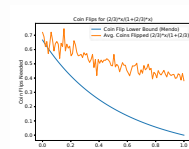
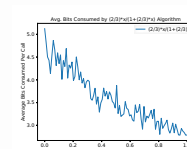
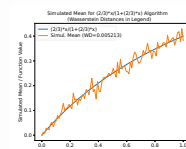
$$(1-x)/\cos(x)$$



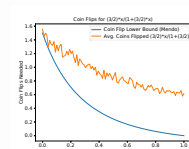
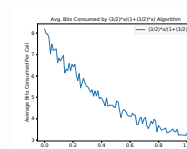
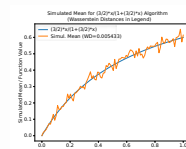
$$(1/3)*x/(1+(1/3)*x)$$



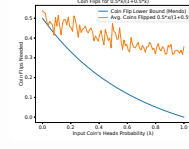
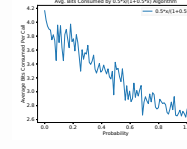
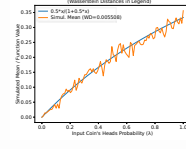
$$(2/3)*x/(1+(2/3)*x)$$



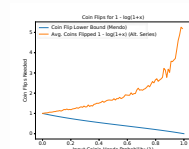
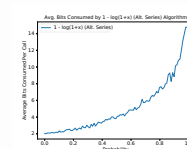
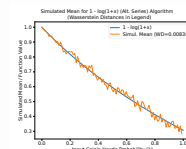
$$(3/2)*x/(1+(3/2)*x)$$



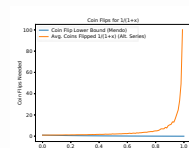
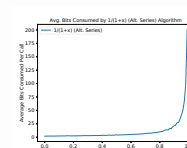
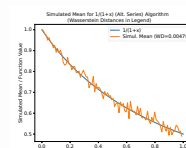
$$0.5*x/(1+0.5*x)$$



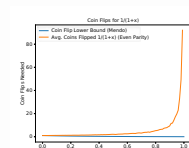
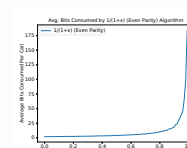
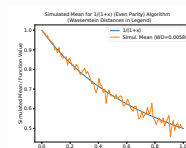
$$1 - \ln(1+x) \text{ (Alt. Series)}$$



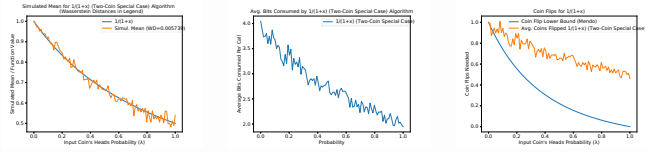
$$1/(1+x) \text{ (Alt. Series)}$$



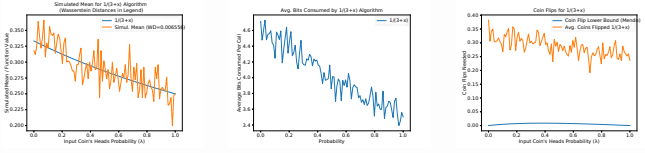
$$1/(1+x) \text{ (Even Parity)}$$



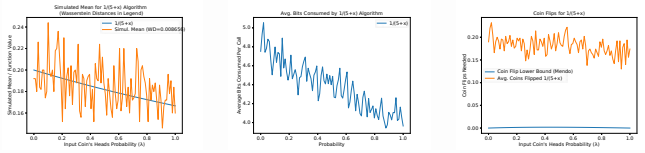
1/(1+x) (Two-Coin Special Case)



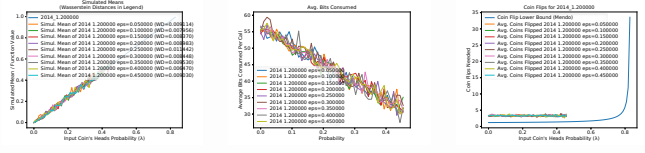
1/(3+x)



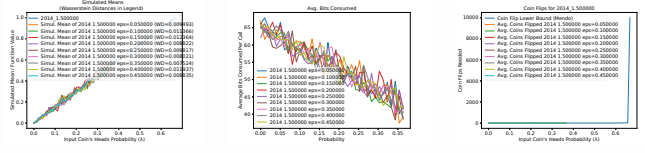
1/(5+x)



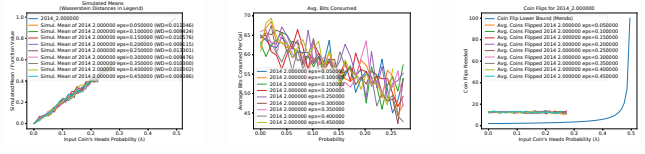
2014 1.200000
eps=0.050000



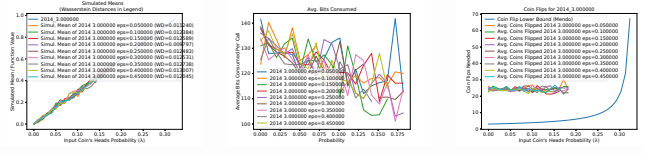
2014 1.500000
eps=0.050000



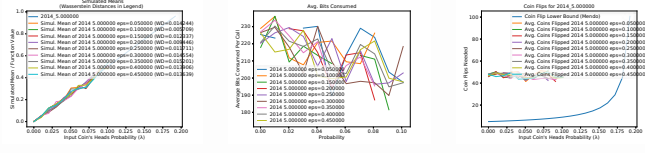
2014 2.000000
eps=0.050000



2014 3.000000
eps=0.050000



2014 5.000000
eps=0.050000



2014 Add. x+0.1

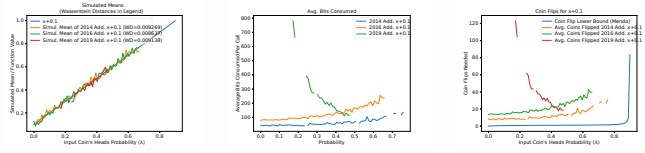


Figure 1 consists of three plots. The left plot, titled 'Dispersed Values (Observations minus Regression)', shows True Values (mm) on the y-axis (0 to 1.5) versus Predicted Values (mm) on the x-axis (0.5 to 0.9). It includes data for 2014-2016 (green dots) and 2017-2019 (red dots), with a blue line representing the regression. The middle plot, titled 'Avg. RMSE Computed', shows Average RMSE (mm) on the y-axis (0 to 10) versus Normalized Error on the x-axis (0.5 to 0.9). It compares the proposed method (green line) with other methods (orange and blue lines). The right plot, titled 'Avg. RMSE for 2017-2019', shows RMSE (mm) on the y-axis (0 to 80) versus Normalized Error on the x-axis (0.5 to 0.9). It compares the proposed method (green line) with other methods (orange and blue lines).

Figure 1 consists of three subplots comparing the proposed method (CUP Filter) with state-of-the-art methods (CUP Filter Lower Bound, CUP Filter, and CUP Filter Upper Bound) across different datasets and AUC values.

Top Left Plot: Stratified Mean AUC (std) vs AUC

This plot shows the stratified mean AUC (std) for different methods across various datasets. The x-axis represents the AUC value, and the y-axis represents the stratified mean AUC (std). The legend indicates the following methods and datasets:

- $\alpha=0.5$ (Blue line)
- 2014 data set (Orange line)
- 2015 data set (Green line)
- 2016 data set (Red line)
- 2017 data set (Yellow line)
- 2018 data set (Cyan line)
- 2019 data set (Magenta line)
- 2020 data set (Dark blue line)

Top Right Plot: Average ROC Curves (std) vs AUC

This plot shows the average ROC curves (std) for different methods across various datasets. The x-axis represents the AUC value, and the y-axis represents the average ROC curves (std). The legend indicates the following methods and datasets:

- 2014 data set (Orange line)
- 2015 data set (Green line)
- 2016 data set (Red line)
- 2017 data set (Yellow line)
- 2018 data set (Cyan line)
- 2019 data set (Magenta line)
- 2020 data set (Dark blue line)

Bottom Plot: CUP Filter for AUC=0.5 vs CUP Filter for AUC=0.5

This plot shows the CUP Filter for AUC=0.5 for different methods across various datasets. The x-axis represents the CUP Filter for AUC=0.5, and the y-axis represents the CUP Filter for AUC=0.5. The legend indicates the following methods and datasets:

- CUP Filter Lower Bound (Blue line)
- CUP Filter (Orange line)
- CUP Filter Upper Bound (Green line)
- 2014 data set (Orange line)
- 2015 data set (Green line)
- 2016 data set (Red line)
- 2017 data set (Yellow line)
- 2018 data set (Cyan line)
- 2019 data set (Magenta line)
- 2020 data set (Dark blue line)

[illegible]

Figure 1 consists of three subplots labeled (a), (b), and (c), each showing performance metrics for the proposed model across different input data sets.

(a) ROC curve for v1.5: The plot shows the True Positive Rate (TPR) on the y-axis (0.0 to 1.0) versus the False Positive Rate (FPR) on the x-axis (0.0 to 1.0). A diagonal line represents random performance. The proposed model's performance is shown as a red line, which is significantly above the diagonal, indicating good performance. The legend lists the following data sets:

- Input: 2014 v1.5, v1.5 (train=0.8, test=0.2)
- Input: 2014 v1.5, v1.5 (train=0.7, test=0.3)
- Input: 2014 v1.5, v1.5 (train=0.6, test=0.4)
- Input: 2014 v1.5, v1.5 (train=0.5, test=0.5)
- Input: 2014 v1.5, v1.5 (train=0.4, test=0.6)
- Input: 2014 v1.5, v1.5 (train=0.3, test=0.7)
- Input: 2014 v1.5, v1.5 (train=0.2, test=0.8)
- Input: 2014 v1.5, v1.5 (train=0.1, test=0.9)

(b) Average ROC curve for v1.5: The plot shows the Average ROC Curve on the y-axis (0.0 to 1.0) versus the Input Data Set Probability on the x-axis (0.0 to 1.0). The proposed model's performance is shown as a red line, which is significantly above the diagonal, indicating good performance. The legend lists the following data sets:

- Input: 2014 v1.5, v1.5 (train=0.8, test=0.2)
- Input: 2014 v1.5, v1.5 (train=0.7, test=0.3)
- Input: 2014 v1.5, v1.5 (train=0.6, test=0.4)
- Input: 2014 v1.5, v1.5 (train=0.5, test=0.5)
- Input: 2014 v1.5, v1.5 (train=0.4, test=0.6)
- Input: 2014 v1.5, v1.5 (train=0.3, test=0.7)
- Input: 2014 v1.5, v1.5 (train=0.2, test=0.8)
- Input: 2014 v1.5, v1.5 (train=0.1, test=0.9)

(c) Cost-Peak Ratio for v1.5: The plot shows the Cost-Peak Ratio on the y-axis (0.0 to 1.0) versus the Input Data Set Probability on the x-axis (0.0 to 1.0). The proposed model's performance is shown as a red line, which is significantly above the diagonal, indicating good performance. The legend lists the following data sets:

- Input: 2014 v1.5, v1.5 (train=0.8, test=0.2)
- Input: 2014 v1.5, v1.5 (train=0.7, test=0.3)
- Input: 2014 v1.5, v1.5 (train=0.6, test=0.4)
- Input: 2014 v1.5, v1.5 (train=0.5, test=0.5)
- Input: 2014 v1.5, v1.5 (train=0.4, test=0.6)
- Input: 2014 v1.5, v1.5 (train=0.3, test=0.7)
- Input: 2014 v1.5, v1.5 (train=0.2, test=0.8)
- Input: 2014 v1.5, v1.5 (train=0.1, test=0.9)

Figure 1 consists of three subplots labeled (a), (b), and (c).
 Subplot (a) is a scatter plot titled "Scatter Plot of Predicted vs. Actual Mean (Covariance)". The x-axis is "Input C₁ Mean Probability (%)" ranging from 0.0 to 0.5. The y-axis is "Predicted Mean (Covariance)" ranging from 0.0 to 1.0. Data points are colored circles representing different input mean probabilities: 0.0, 0.1, 0.2, 0.3, 0.4, and 0.5. A red diagonal line represents the ideal prediction. The points closely follow this line.
 Subplot (b) is a line plot titled "Avg. Mean Squared Error (MSE) vs. Probability". The x-axis is "Probability" ranging from 0.0 to 0.5. The y-axis is "Average MSE (Covariance)" ranging from 0.0 to 0.05. Multiple lines represent different input mean probabilities: 0.0 (blue), 0.1 (orange), 0.2 (green), 0.3 (red), 0.4 (purple), and 0.5 (brown). The MSE generally increases with probability, with the 0.5 input mean probability showing the highest MSE.
 Subplot (c) is a line plot titled "Cost Ratio vs. Probability". The x-axis is "Input C₁ Mean Probability (%)" ranging from 0.0 to 0.5. The y-axis is "Cost Ratio" ranging from 0 to 100. Multiple lines represent different input mean probabilities: 0.0 (blue), 0.1 (orange), 0.2 (green), 0.3 (red), 0.4 (purple), and 0.5 (brown). The cost ratio is highest at low probabilities and decreases as probability increases, with the 0.5 input mean probability showing the lowest cost ratio.

Figure 1 consists of two subplots, (a) and (b), showing the performance of the proposed model.

Subplot (a) is a Receiver Operating Characteristic (ROC) curve. The x-axis is labeled "Input Coir's Weave Probability (x)" and ranges from 0.00 to 0.40. The y-axis is labeled "Detection Rate (True Positive Rate)" and ranges from 0.00 to 1.00. The plot shows several curves for different models:

- Proposed (blue line): This curve is the highest, indicating the best performance.
- Proposed + 2018-2019 (orange line): This curve is slightly below the proposed model.
- Proposed + 2018-2019 + 2020-2021 (green line): This curve is below the previous one.
- Proposed + 2018-2019 + 2020-2021 + 2022-2023 (red line): This curve is below the previous one.
- Proposed + 2018-2019 + 2020-2021 + 2022-2023 + 2024-2025 (purple line): This curve is the lowest among the proposed model variants.
- Random (black line): This is a diagonal line from (0,0) to (1,1), representing a random classifier.

Subplot (b) is a line graph showing the Average ROC Curve for the proposed model across different input data sets. The x-axis is labeled "Probability" and ranges from 0.00 to 0.50. The y-axis is labeled "Average ROC Curve" and ranges from 0.00 to 1.00. The plot shows several curves for different input data sets:

- Coir Weave (blue line): This curve is the highest, indicating the best performance.
- Coir Weave + 2018-2019 (orange line): This curve is slightly below the Coir Weave model.
- Coir Weave + 2018-2019 + 2020-2021 (green line): This curve is below the previous one.
- Coir Weave + 2018-2019 + 2020-2021 + 2022-2023 (red line): This curve is below the previous one.
- Coir Weave + 2018-2019 + 2020-2021 + 2022-2023 + 2024-2025 (purple line): This curve is the lowest among the proposed model variants.
- Random (black line): This is a diagonal line from (0,0) to (1,1), representing a random classifier.

The figure consists of three vertically stacked plots sharing a common x-axis representing input correlation probability from 0.00 to 0.15.

- Top Plot:** Average Normalized Error Rate vs. Input Correlation Probability. The y-axis ranges from 0.00 to 1.00. All methods show an increasing error rate as correlation increases. The proposed method (blue) generally has the lowest error rate across most correlations.
- Middle Plot:** Average Coverage Rate (%) vs. Probability. The y-axis ranges from 0 to 100. Most methods maintain coverage near 100%, while the proposed method (blue) starts lower at ~80% and decreases slightly as correlation increases.
- Bottom Plot:** Cost per Bit vs. Input Correlation Probability. The y-axis ranges from 0 to 160. The proposed method (blue) shows a sharp increase in cost starting around 0.10 correlation, reaching over 160 at 0.15 correlation. Other methods remain relatively flat or show much less dramatic increases.

Figure 1 consists of two plots, (a) and (b), comparing the proposed model with existing models.

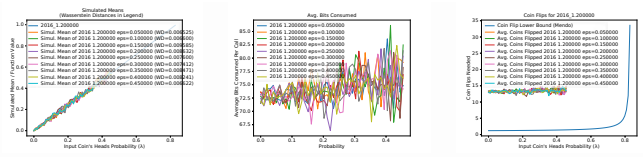
Plot (a) shows the Standard Error of the Mean (SEM) on the y-axis (ranging from 0.00 to 0.02) versus Input Concentration (mg/L) on the x-axis (ranging from 0.00 to 0.60). The legend includes:

- Prop. (red line)
- Linear (blue line)
- Linear + 1st Order (green line)
- Linear + 2nd Order (purple line)
- Linear + 3rd Order (orange line)
- Linear + 4th Order (brown line)
- Linear + 5th Order (pink line)
- Linear + 6th Order (grey line)
- Linear + 7th Order (light blue line)
- Linear + 8th Order (light green line)
- Linear + 9th Order (light orange line)
- Linear + 10th Order (light purple line)
- Linear + 11th Order (light brown line)
- Linear + 12th Order (light pink line)
- Linear + 13th Order (light grey line)
- Linear + 14th Order (light blue line)
- Linear + 15th Order (light green line)
- Linear + 16th Order (light orange line)
- Linear + 17th Order (light purple line)
- Linear + 18th Order (light brown line)
- Linear + 19th Order (light pink line)
- Linear + 20th Order (light grey line)

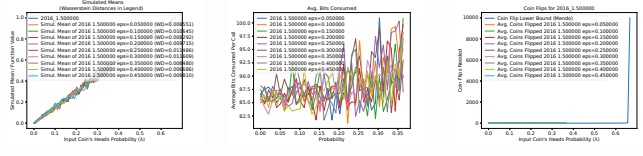
Plot (b) shows the Average Bio-Concentration Factor (BCF) on the y-axis (ranging from 0 to 200) versus Input Concentration (mg/L) on the x-axis (ranging from 0.00 to 0.60). The legend includes:

- Prop. (red line)
- Linear (blue line)
- Linear + 1st Order (green line)
- Linear + 2nd Order (purple line)
- Linear + 3rd Order (orange line)
- Linear + 4th Order (brown line)
- Linear + 5th Order (pink line)
- Linear + 6th Order (grey line)
- Linear + 7th Order (light blue line)
- Linear + 8th Order (light green line)
- Linear + 9th Order (light orange line)
- Linear + 10th Order (light purple line)
- Linear + 11th Order (light brown line)
- Linear + 12th Order (light pink line)
- Linear + 13th Order (light grey line)
- Linear + 14th Order (light blue line)
- Linear + 15th Order (light green line)
- Linear + 16th Order (light orange line)
- Linear + 17th Order (light purple line)
- Linear + 18th Order (light brown line)
- Linear + 19th Order (light pink line)
- Linear + 20th Order (light grey line)

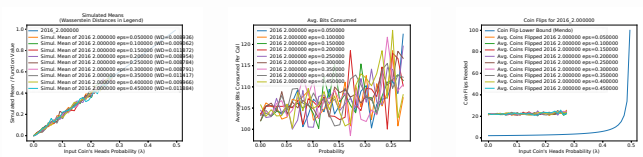
```
2016 1.200000
eps=0.050000
```



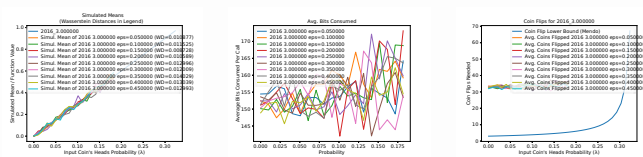
```
2016 1.500000
eps=0.050000
```



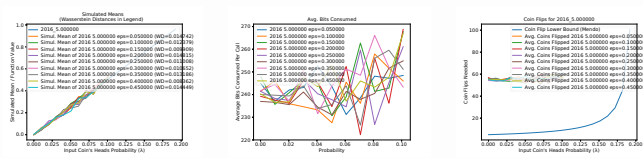
```
2016 2.000000
eps=0.050000
```



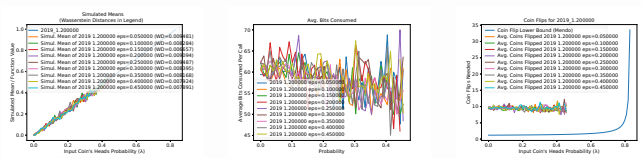
```
2016 3.000000
eps=0.050000
```



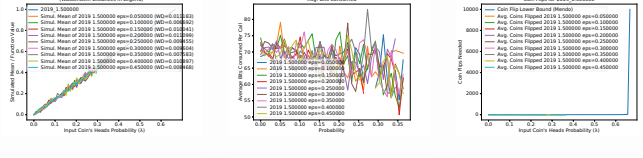
```
2016 5.000000
eps=0.050000
```



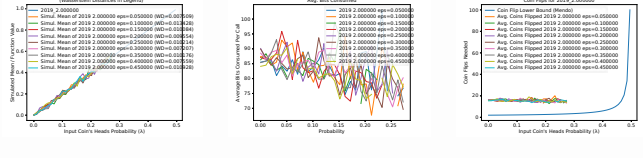
```
2019 1.200000
eps=0.050000
```



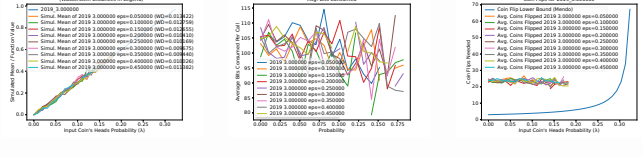
```
2019 1.500000
eps=0.050000
```



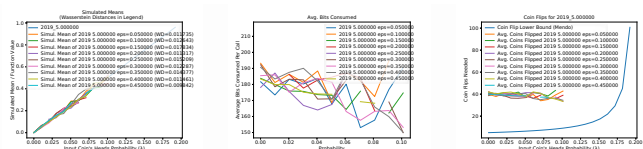
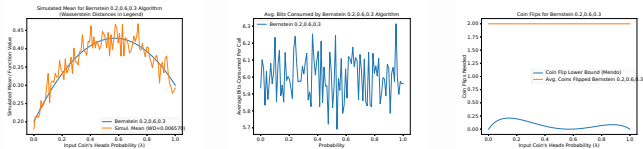
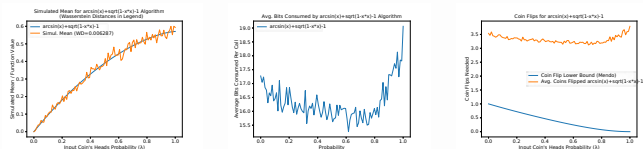
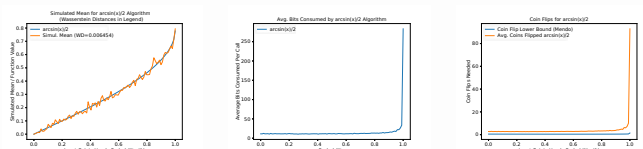
```
2019 2.000000
eps=0.050000
```



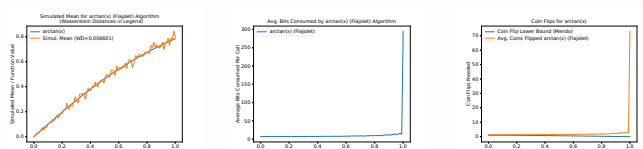
2019 3.000000
eps=0.050000



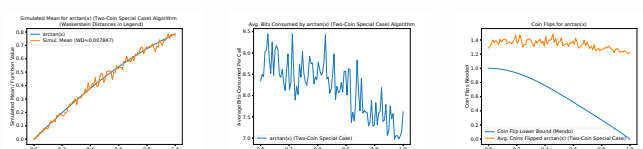
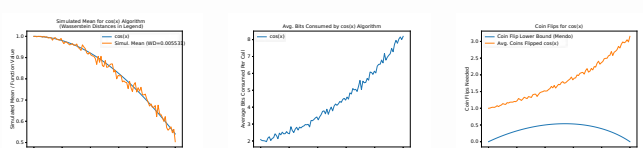
```
2019 5.000000
eps=0.050000
```

Bernstein
0.2,0.6,0.3
$$\arcsin(x) + \sqrt{1-x^2} - 1$$
 $\arcsin(x)/2$ 

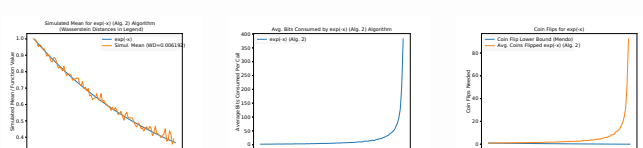
arctan(x)
(Flajolet)



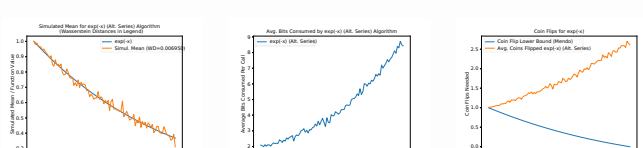
arctan(x) (Two-Coin Special Case)

 $\cos(x)$ 

$\exp(-x)$ (Alg. 2)



exp(-x) (Alt.
Series)



Estimated Mean for each of (Proposed) Algorithm

Standard Error (times) Risk

Logarithmic Coefficient (β)

Exact Mean (0.001-0.005)

Prop. Mean (0.001-0.005)

Average Coefficient (β)

Logarithmic Coefficient (β)

CPU Time (sec)

Logarithmic Coefficient (β)

Exact Mean (0.001-0.005)

Prop. Mean (0.001-0.005)

Figure 10 consists of three subplots:

- Left Plot:** Scalability of the proposed algorithm. The x-axis is 'Input Data Size (Number of Nodes)' ranging from 0.0 to 1.0. The y-axis is 'Scalability of the Proposed Algorithm' ranging from 0.0 to 1.0. The plot shows two curves: 'Proposed' (blue line with circles) and 'Baseline' (orange line with circles). Both curves start at 1.0 and decrease as the input data size increases, with the proposed algorithm maintaining higher scalability than the baseline.
- Middle Plot:** Avg. Bits Consumed by the proposed algorithm. The x-axis is 'Input Data Size' ranging from 0.0 to 1.0. The y-axis is 'Average Bits Consumed' ranging from 0.0 to 1.0. The plot shows two curves: 'Proposed' (blue line with circles) and 'Baseline' (orange line with circles). Both curves start at 0.0 and increase as the input data size increases, with the proposed algorithm consuming fewer bits than the baseline.
- Right Plot:** Gap Error for input data size. The x-axis is 'Input Data Size (Number of Nodes)' ranging from 0.0 to 1.0. The y-axis is 'Gap Error' ranging from 0.0 to 1.0. The plot shows two curves: 'Proposed' (blue line with circles) and 'Baseline' (orange line with circles). Both curves start at 0.0 and increase as the input data size increases, with the proposed algorithm showing a lower gap error than the baseline.

Serialized Mean for $\log_2(x+1)$ (Pigeon) Algorithm
Serialized Mean for $\log_2(x+1)$ (Proposed) Algorithm

Serialized Mean for $\log_2(x+1)$ (Pigeon) Algorithm
Serialized Mean for $\log_2(x+1)$ (Proposed) Algorithm

Cost Ratio for $\log_2(x+1)$

Cost-Flip Greedy
Avg. Cost-Flip Greedy

Figure 1 consists of three subplots labeled (a), (b), and (c).
 Subplot (a) is titled 'Scalability: Disabled flow capacity (kA) vs. Number of nodes'. The x-axis is 'Number of nodes' from 0.0 to 1.0. The y-axis is 'Disabled flow capacity (kA)' from 0.0 to 1.0. It shows two curves: 'points 1-10' (blue line with dots) and 'points 100-1000' (orange line with dots). Both curves start at (0,0) and increase monotonically, with the orange curve being slightly higher than the blue curve.
 Subplot (b) is titled 'Convergence: Average flow capacity (kA) vs. Iteration number'. The x-axis is 'Iteration number' from 0.0 to 1.0. The y-axis is 'Average flow capacity (kA)' from 0.0 to 1000.0. It shows two curves: 'points 1-10' (blue line with dots) and 'points 100-1000' (orange line with dots). Both curves start at approximately 1000 kA and drop sharply to near zero by iteration 0.2, remaining there until iteration 1.0.
 Subplot (c) is titled 'Error: Error (kA) vs. Iteration number'. The x-axis is 'Iteration number' from 0.0 to 1.0. The y-axis is 'Error (kA)' from 0 to 400. It shows two curves: 'points 1-10' (blue line with dots) and 'points 100-1000' (orange line with dots). Both curves start at approximately 400 kA and drop sharply to near zero by iteration 0.2, remaining there until iteration 1.0.

Figure 10 consists of three subplots comparing the performance of the proposed algorithm (grey line) with the state-of-the-art algorithm (orange line) for the $p=213$ case. The x-axis for all plots is the Signal-to-noise ratio (SNR) ranging from 0.0 to 1.0.

- Left Plot:** Dryad Error ($1/2 ||x - \hat{x}||_2^2$) vs. SNR. The y-axis ranges from 0.0 to 1.8. The legend indicates 'Proposed (213)' (grey line) and 'Oracle (Mean (0.000000))' (orange line). Both lines show an increasing trend, with the proposed algorithm's error being slightly lower than the oracle's error for SNR > 0.5.
- Middle Plot:** Average RMSE ($||x - \hat{x}||_2$) vs. SNR. The y-axis ranges from 0.0 to 0.6. The legend indicates 'Proposed (213)' (grey line). The RMSE increases with SNR, starting around 0.25 at SNR=0.0 and reaching approximately 0.55 at SNR=1.0.
- Right Plot:** Case File Loss ($||x - \hat{x}||_2$) vs. SNR. The y-axis ranges from 0.00 to 2.00. The legend indicates 'Case File Loss (Oracle (0.000000))' (orange line) and 'Proposed (213)' (grey line). The proposed algorithm's loss is significantly lower than the oracle's loss for SNR < 0.5, but becomes slightly higher for SNR > 0.5.

Figure 1 consists of three subplots showing the performance of the proposed algorithm on the 2014 dataset. The x-axis for all plots is 'SE' (Standard Error), ranging from 0.0 to 1.0.

- Left Subplot:** Titled 'Simulated SE for points 243-244 Algorithm'. The y-axis is 'Standard Error (SE) vs. SE'. It shows two curves: a blue line for 'points 243' and an orange line for 'points 244'. Both curves start at (0,0) and increase monotonically, reaching 1.0 at SE=1.0.
- Middle Subplot:** Titled 'Avg. SE Computed by points 243-244 Algorithm'. The y-axis is 'Average SE Computed by points 243-244 Algorithm'. It shows two curves: a blue line for 'points 243' and an orange line for 'points 244'. Both curves start at (0,1.0) and decrease monotonically, reaching 0.0 at SE=1.0.
- Right Subplot:** Titled 'CDF Plots for points 243-244'. The y-axis is 'CDF vs. SE'. It shows two curves: a blue line for 'points 243' and an orange line for 'points 244'. Both curves start at (0,1.0) and decrease monotonically, reaching 0.0 at SE=1.0.

Figure 1 consists of three subplots labeled (a), (b), and (c), comparing the performance of the proposed algorithm (points 3,4) against a baseline (points 3,4) and a baseline (points 3,4).

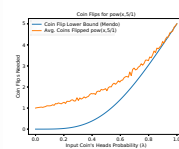
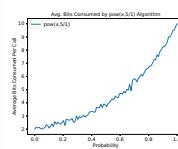
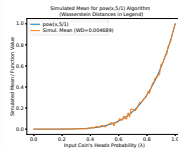
(a) Standard Error (log scale) vs. Capital Asset Turnover Ratio (X). The y-axis ranges from 0.00 to 1.00 on a log scale. The x-axis ranges from 0.00 to 1.00. The legend indicates: points 3,4 (blue line), points 3,4 (orange line), and Baseline (points 3,4) (red line). The proposed algorithm (points 3,4) shows a significantly lower standard error compared to the baseline.

(b) Avg. Bits Consumed (log scale) vs. Precision. The y-axis ranges from 0.00 to 1.00 on a log scale. The x-axis ranges from 0.00 to 1.00. The legend indicates: points 3,4 (blue line). The proposed algorithm (points 3,4) shows a significantly lower average bits consumed compared to the baseline.

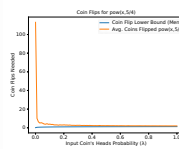
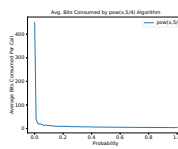
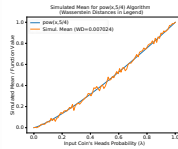
(c) Cost Functions vs. Capital Asset Turnover Ratio (X). The y-axis ranges from 0.00 to 1.00. The x-axis ranges from 0.00 to 1.00. The legend indicates: Cost Fcn Upper Bound (blue line), Cost Fcn Upper Bound (orange line), and Cost Fcn Upper Bound (red line). The proposed algorithm (points 3,4) shows a significantly lower cost function value compared to the baseline.

Figure 1 consists of two plots. Plot (a) is an ROC curve titled 'Simulated Mean for pnorm, K5, Algorithm Comparison: Simulated (0.5000000)' and 'Empirical Mean (0.5000000)'. The x-axis is 'True Positive Fraction (Rate)' and the y-axis is 'Simulated True Fraction (Rate)', both ranging from 0.0 to 1.0. A diagonal line represents random performance. The 'Empirical Mean' (orange line) and 'Simulated Mean' (blue line) are nearly identical and follow the diagonal. Plot (b) is titled 'Avg. Bits Consumed by pnorm, K5, Algorithm'. The x-axis is 'Input Class Probability (a)' from 0.0 to 1.0, and the y-axis is 'Average Bits Consumed by Algorithm' from 0.0 to 10.0. The 'pnorm, K5' algorithm (blue line) shows a sharp decrease in bits consumed as input class probability increases, starting at approximately 9.5 bits for probability 0.0 and reaching about 0.5 bits at probability 1.0. The legend for both plots includes 'pnorm, K5' (blue line) and 'Emp. Mean' (orange line).

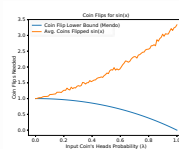
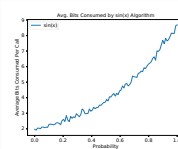
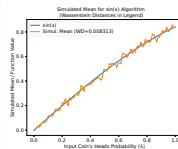
pow(x,5/1)



pow(x,5/4)



sin(x)



sqrt(x)

