Approximating the Permanent with Deep Rejection Sampling
Supplement

Juha Harviainen
University of Helsinki
juha.harviainen@helsinki.fi

Antti Röyskö
ETH Zürich
aroeyskoe@student.ethz.ch

Mikko Koivisto
University of Helsinki
mikko.koivisto@helsinki.fi

Contents

35th Conference on Neural Information Processing Systems (NeurIPS 2021).

A Algorithms

A.1 Gamma Bernoulli acceptance sheme

For any $\epsilon \in (0, 3/4)$, the Gamma Bernoulli acceptance scheme, GBAS, due to Huber [1] yields an (ϵ, δ)-approximation requiring at most $\lceil \psi^*(\epsilon, \delta) \rceil$ accepted draws, with

$$\psi^*(\epsilon, \delta) := 2\big(1 - (4/3)\epsilon\big)^{-1} \epsilon^{-2} \ln(2/\delta)\,.$$

The estimate is obtained by normalizing the number of accepts minus one by a Gamma distributed variable, which is generated as a sum of random variables that follow $\mathrm{Exp}(1)$ distribution:

Function GBAS-ESTIMATE$(\Omega, u, \epsilon, \delta)$

G1 $k \leftarrow \lceil \psi^*(\epsilon, \delta) \rceil$, $t \leftarrow 0$, $s \leftarrow 0$

G2 Repeat $t \leftarrow t +$ EXP(1), $s \leftarrow s +$ SAMPLE(Ω, u) until $s = k$

G2 Return $u(\Omega) \cdot (k-1)/t$

In fact, supposing one can evaluate the cumulative distribution function of $\mathrm{Gamma}(k, k-1)$ for any positive integer k, one can replace the upper bound above by the exact minimizing value in step G1:

$$k \leftarrow \min\big\{k : Z \sim \mathrm{Gamma}(k, k-1), \Pr(|Z-1| > \epsilon) < \delta\big\}\,.$$

For instance, with $\epsilon = 0.1$ and $\delta = 0.05$ we get that $k = 388$ accepted draws suffice.

A.2 Preprocessing: Sinkhorn balancing

The rejection sampling method of Huber and Law [2] makes the matrices nearly doubly stochastic, meaning that all row and column sums of the matrix are close to 1. There are multiple ways of approaching the problem of making a matrix nearly doubly stochastic, and we follow the methodology of Sullivan and Beichl [5]. They use Sinkhorn balancing, a method in which we alternate between dividing each row vector and each column vector by their corresponding sums. If for every positive entry there exists a permutation of positive weight to which it belongs to, the Sinkhorn balancing is guaranteed to converge linearly in the number of iterations [4].

This property can be obtained by interpreting the matrix as a (weighted) adjacency matrix of a bipartite graph and running Tassa's algorithm [6] on it, which removes the edges that are not part of any perfect matching. According to Sullivan and Beichl, running n^2 iterations of Sinkhorn balancing after this tends to be sufficient for getting the matrix close enough to being doubly stochastic. Finally, Huber and Law divide each row vector by their largest value. This last step can never increase the ratio of the Huber–Law bound and the exact value of the permanent. For the Brouwer–Schrijver bound, there is no difference.

A.3 Incremental computation of the Huber–Law and Brouwer–Schrijver bounds

The rejection sampler based on the Huber–Law bound is straightforward to implement to work in $O(n^2)$ time. As we proceed, we maintain the row sums of a matrix $A = (a_{ij})$ in an array $[r_1, r_2, \ldots, r_n]$. When we enter a column j, we decrease the row sums by their corresponding entries in the column. Denote the matrix obtained from A by zeroing all entries on the ith row and the jth column except the entry that is on the ith row and the jth column by $f(A, i, j)$. Then, each upper bound $U^{\mathrm{HL}}(f(A, i, j))$ can be written as

$$a_{ij} \cdot \prod_{s=1, s\neq i}^{n} \frac{h(r_s)}{e},$$

where h is the function used in the Huber–Law bound. All bounds for a column j can be computed in $O(n)$ time by precomputing products $\prod_{s=1}^{k} h(r_s)/e$ and $\prod_{s=k}^{n} h(r_s)/e$ for all $k \in N$. After using these upper bounds for picking a row i, we replace A with $f(A, i, j)$ and r_i with a_{ij}. A similar method is applied in implementing the rejection sampler based on the Brouwer–Schrijver bound. We will show that, for a matrix $A = (a_{ij})$, we can compute the values $U^{\mathrm{SS}}(f(A, i, j))$ for all $i, j \in N$

in $O(n^2)$ time after some preprocessing. Assuming that there is always a column on which the Brouwer–Schrijver bound is nesting, we can pick the column with the smallest sum of upper bounds and use that column to sample a row–column pair (i, j). After that, if we do not reject the sample, we apply the same procedure on the matrix obtained from A by removing the ith row and the jth column. If the sample does not get not rejected whilst running the procedure repeatedly, we will eventually end up with a positive 1×1 matrix at which point we accept the sample. Therefore, the total time complexity per trial then becomes $O(n^3)$ given that the assumption holds.

The idea for computing the upper bounds is to maintain for each row i an array of columns j sorted in decreasing order by the entries a_{ij}. Letting $\pi_i(t)$ to denote the tth column, write $a^*_{it} := a_{ij}$ when $j = \pi_i(t)$. Write also $\delta_k := \gamma(k) - \gamma(k-1)$ for short. We observe that the n^2 values $U(f(A, i, j))$ can be written as

$$a^*_{it} \cdot \prod_{s=1, s\neq i}^{n} \Big(\sum_{k=1}^{t-1} a^*_{sk}\delta_k + \sum_{k=t+1}^{n} a^*_{sk}\delta_{k-1} \Big), \quad \text{for } i, t = 1, 2, \ldots, n\,.$$

For each fixed s, the $2n$ sums, for $t = 1, 2, \ldots, n$, are computed in time $O(n)$ by using the values $\pi_s(t)$ and sum arrays. To maintain the values $\pi_s(t)$, we take the entries in each row of the matrix and sort them at the beginning of the sampling, and after we delete a row and a column from the matrix, it is simple to update the values in $O(n^2)$ time. With these values at hand for all s and t, the n^2 products are computed in time $O(n^2)$ in a similar manner as with the Huber–Law bound. As the sorting needs to be done only once and we need to compute $O(n^2)$ upper bounds for each subproblem, maintaining the values does not increase the time complexity.

B Proofs

B.1 Proof of Lemma 5

Suppose $p = mn^{-2}$. Then the mean of the binomial variable M equals m. Since the mean is an integer, M has a unique median equalling the mean m. We will use the fact that for any nonnegative random variable X, a median m_X is at most twice the mean. Indeed, by Markov's inequality,

$$\frac{1}{2} \leq \Pr(X \geq m_X) \leq \frac{\mathbf{E}[X]}{m_X}.$$

Consider the random variable $X := \mathbf{E}[U \mid M]$. Since $\mathbf{E}[U \mid M = i]$ is increasing in i, there is a median m_X that is at least $\mathbf{E}[U \mid M = m]$, whence $\mathbf{E}[U \mid M = m] \leq 2\,\mathbf{E}[U]$.

B.2 Proof of Lemma 6

We will bound the expected value of the depth-d Huber–Law upper bound $U := U^{\mathrm{HL}}_d(A)$ of a random $(0,1)$-matrix A. Write $t := n - d$. For any $t \times t$ submatrix A_{IK} of A, we have $U^{\mathrm{HL}}_d(A_{IK}) = \mathrm{e}^{-t} \prod_{i\in I} h(r_i)$, where r_i is the number of 1s in the row A_{iK}. Thus, by linearity of expectation and independence of the entries, we get

$$\mathbf{E}[U] = \binom{n}{d} d!\, p^d\, \mathrm{e}^{-t}\, \mathbf{E}[h(X)]^t\,,$$

where X is a binomial random variable with t trials and success probability p. It remains to show that $h(r)$ is concave in $[0, \infty)$, for then Jensen's inequality gives us $\mathbf{E}[h(X)] \leq h\big(\mathbf{E}[X]\big) = h(tp)$.

To see that h is concave, recall that $h(r) = 1 + (\mathrm{e} - 1)r$ if $r \leq 1$ and $h(r) = \mathrm{e} - 1 + r + \frac{1}{2}\ln r$ if $r \geq 1$. Clearly, h is separately concave in $[0, 1]$ and in $[1, \infty)$. Therefore, it suffices to show that the first derivative $h'(r)$ is at most $\mathrm{e} - 1$ for all $r \geq 1$. Calculation gives $h'(r) = 1 + \frac{1}{2r} \leq \frac{3}{2} < \mathrm{e} - 1$.

B.3 Proof of Lemma 7

By Lemma 4,

$$\mathbf{E}[V \mid M = m] = n! \left(\frac{m}{n^2}\right)^n \exp\Big\{ -\frac{n^2}{2m} + \frac{1}{2} - O\Big(\frac{n^3}{m^2}\Big)\Big\}.$$

On the other hand, combining Lemmas 5 and 6 gives

$$\mathbf{E}[U \mid M = m] \le 2\binom{n}{d} d!\, p^d\, \mathrm{e}^{d-n}\, h\big((n-d)p\big)^{n-d},$$

where $p = mn^{-2}$. Writing $t := n - d$, taking the ratio, and simplifying yields

$$\rho := \frac{\mathbf{E}[U \mid M = m]}{\mathbf{E}[V \mid M = m]} \le \frac{2\, h(tp)^t}{t!\, p^t\, \mathrm{e}^t} \cdot \exp\Big\{\frac{1}{2p} - \frac{1}{2} + O\Big(\frac{n^3}{m^2}\Big)\Big\}.$$

Using the Stirling bound $t! \ge (2\pi t)^{-1/2} t^t \mathrm{e}^{-t}$, we get

$$\rho \le \frac{2}{\sqrt{2\pi t}} \frac{h(tp)^t}{(tp)^t} \cdot \exp\Big\{\frac{1}{2p} - \frac{1}{2} + O\Big(\frac{n^3}{m^2}\Big)\Big\}.$$

Since $d \le n - p^{-1}$, we have $(n-d)p = tp \ge 1$, and thus

$$\frac{h(tp)}{tp} = 1 + \frac{2\mathrm{e} - 2 + \ln(tp)}{2tp}.$$

Using $(1 + x/t)^t \le \mathrm{e}^x$ gives

$$\Big(\frac{h(tp)}{tp}\Big)^t \le \exp\Big\{\frac{1}{2p}\big(2\mathrm{e} - 2 + \ln(tp)\big)\Big\} = \big(\mathrm{e}^{2\mathrm{e}-2} pt\big)^{1/(2p)}.$$

Simplifying,

$$\rho \le \big(\pi \mathrm{e} t/2\big)^{-1/2} \big(\mathrm{e}^{2\mathrm{e}-1} pt\big)^{1/(2p)} \cdot \mathrm{e}^{O(n^3 m^{-2})}.$$

B.4 Proof of Lemma 9

We let $\underline{\Pr}$, $\underline{\mathbf{E}}$, and $\underline{\mathbf{Var}}$ refer, respectively, to the probability measure, the expectation, and the variance conditional on the event $M = m$. Let $0 < \alpha, \beta < 1$.

By Markov's inequality,

$$\underline{\Pr}(U \ge \alpha^{-1}\, \underline{\mathbf{E}}[U]) \le \alpha\,.$$

By Chebyshev's inequality

$$\begin{aligned}
\underline{\Pr}(V \le (1-\beta)\, \underline{\mathbf{E}}[V]) &\le \underline{\Pr}(|V - \underline{\mathbf{E}}[V]| \ge \beta \underline{\mathbf{E}}[V]) \\
&\le \frac{\underline{\mathbf{Var}}[V]}{\beta^2 \underline{\mathbf{E}}[V]^2} \\
&= \beta^{-2}\Big(\frac{\underline{\mathbf{E}}[V^2]}{\underline{\mathbf{E}}[V]^2} - 1\Big) \\
&= \beta^{-2} O(n^3 m^{-2})\,.
\end{aligned}$$

Thus

$$\underline{\Pr}(U \le \alpha^{-1}\, \underline{\mathbf{E}}[U] \text{ and } V \ge (1-\beta)\, \underline{\mathbf{E}}[V]) \ge 1 - \alpha - \beta^{-2} O(n^3 m^{-2})\,.$$

B.5 Argument for Remark 1

Let $c > 1$ and $0 < \delta < 1$ be fixed numbers. Our claim is that, under the conditions of Theorem 2, for all sufficiently large n,

$$\big(\mathrm{e}^{2\mathrm{e}-1}(n-d)p_0\big)^{1/(2p_0)} < c \cdot \big(\mathrm{e}^{2\mathrm{e}-1}(n-d)p\big)^{1/(2p)},$$

where $p_0 := p - n^{-1}\sqrt{2p \ln \delta^{-1}}$. In what follows, write $a := \mathrm{e}^{2\mathrm{e}-1}$ and $b := \sqrt{2 \ln \delta^{-1}}$ for short. To prove the claim, we show that

$$\big(a(n-d)p\big)^{1/p_0 - 1/p} \to 1 \quad \text{as } n \to \infty\,.$$

Since

$$\frac{1}{p_0} - \frac{1}{p} = \frac{p - p_0}{p_0 p} = \frac{b\sqrt{p}}{p(np - b\sqrt{p})} \leq \frac{b}{np^{3/2} - b}$$

and $(n - d)p \leq n$, it suffices to prove that

$$\frac{b \ln(an)}{np^{3/2} - b} \to 0 \quad \text{as } n \to \infty\,.$$

Since a and b are constants and $p \geq n^{-1/2}$ for sufficiently large n, a sufficient condition is

$$\frac{\ln n}{n^{1/4}} \to 0 \quad \text{as } n \to \infty\,.$$

This clearly holds.

C Experimental results

C.1 Test environment

The approximations in the instances of Table 1 of the paper and Figure 1 of the supplemental material were computed on a laptop with one CPU (Kaby Lake R, 1.60 GHz) with a memory limit of 8 GiB. The tests in Figure 2 of the paper and Figure 2 of the supplemental material were run on a computer cluster on one CPU (Sandy Bridge, 2.66 GHz) with a memory limit of 2 GiB. The total running time was limited to two days for each test class and to 4825 seconds for each test instance. The time limit was chosen so that if the computation does not end in roughly 4825 seconds, the time required for getting a $(0.1, 0.05)$-approximation of the permanent would be over eight hours.

C.2 Comparison of AdaPart implementations

We compare our AdaPart-d using the depth-0 bound and the original implementation AdaPart of the AdaPart scheme by Kuck et al. [3] in Figure 1 by estimating the expected running times for getting a $(0.1, 0.05)$-approximation. As both algorithms use the same permanental upper bound, the differences in the running times are mostly due to implementation details. Based on our experimentation, AdaPart-0 is 1–2 orders of magnitude faster than AdaPart, so we used only our implementation in further testing.

C.3 Godsil–Gutman type estimators

To obtain an (ϵ, δ)-approximation of the permanent using the Godsil–Gutman type estimators, one can use the median-of-means trick. This consists of doing $O(\ln \delta^{-1})$ experiments in each of which we draw $O(\epsilon^{-2} c^{n/2})$ samples and take their mean. The value $c^{n/2}$ is the critical ratio of the estimator,

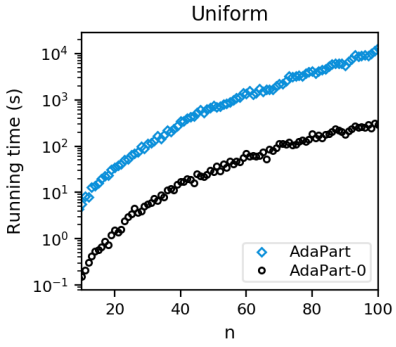

Figure 1: Estimates of expected running times on random instances for both AdaPart implementations.

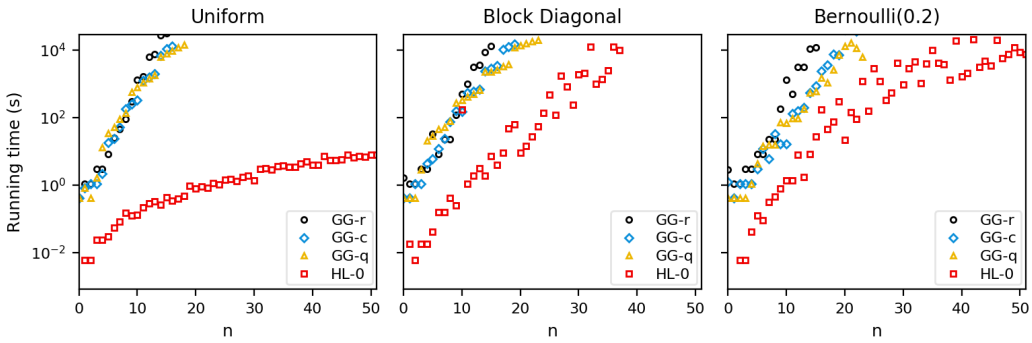

Figure 2: Estimates of expected running times for Godsil–Gutman type indicators and HL-0.

and it is at most $3^{n/2}$, $2^{n/2}$, and $(3/2)^{n/2}$ for reals, complex numbers, and quaternions, respectively. Finally, the median of the means is an (ϵ, δ)-approximation.

For computing the determinant, we used $O(n^3)$-time Gaussian elimination despite the fact that there are faster algorithms for that. Because the exponential critical ratio dominates the running time of the estimator, it seemed extremely unlikely that implementing an algorithm for computing the determinant with a slightly smaller time complexity would have affected the results significantly. We call our implementations of the Godsil–Gutman type estimators based on reals, complex numbers, and quaternions GG-r, GG-c, and GG-q, respectively, and they have been implemented in C++ like our other implementations.

We drew 65 samples to estimate the expected running time of obtaining a $(0.1, 0.05)$-approximation. These were compared to the expected running time of the HL-0 model, which performed the worst in general of our rejection sampling models. The running times are plotted in Figure 2. From these results, it is evident that the Godsil–Gutman estimators are impractical for approximating the permanent even when n is close to 20, while the rejection samplers can usually go much further.

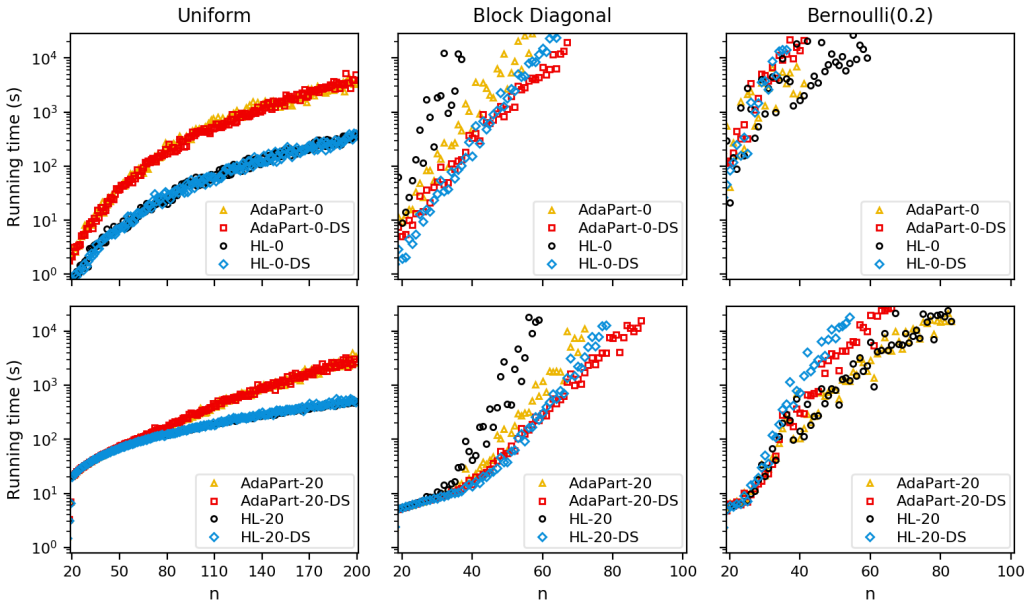

Figure 3: Estimates of expected running times for rejection samplers.

C.4 Comparison of rejection samplers

Figure 3 contains estimates of expected running times for ADAPART-d and HL-d grouped by the depth of the bound. The data is the same as in Figure 2 of the paper, but it is plotted so that the models of the same depth can be compared more easily. In *Uniform*, HL-d beats ADAPART-d by an order of magnitude and preprocessing does not have any significant effect. In the two other classes, ADAPART-d performs better than HL-d. Preprocessing appears helpful for instances of *Block Diagonal* and harmful for *Bernoulli*(0.2).

References

[1] Mark Huber. A Bernoulli mean estimate with known relative error distribution. *Random Struct. Algorithms*, 50(2):173–182, 2017.

[2] Mark Huber and Jenny Law. Fast approximation of the permanent for very dense problems. In *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2008*, pages 681–689. Society for Industrial and Applied Mathematics, 2008.

[3] Jonathan Kuck, Tri Dao, Hamid Rezatofighi, Ashish Sabharwal, and Stefano Ermon. Approximating the permanent by sampling from adaptive partitions. In *Advances in Neural Information Processing Systems 32*, pages 8860–8871. Curran Associates, Inc., 2019.

[4] George W. Soules. The rate of convergence of Sinkhorn balancing. *Linear Algebra Appl.*, 150:3–40, 1991.

[5] Francis Sullivan and Isabel Beichl. Permanents, α-permanents and Sinkhorn balancing. *Comput. Stat.*, 29(6):1793–1798, 2014.

[6] Tamir Tassa. Finding all maximally-matchable edges in a bipartite graph. *Theor. Comput. Sci.*, 423:50–58, 2012.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] See Section 4 and Section 5 of the paper.
   (b) Did you describe the limitations of your work? [Yes] See Section 4 and Section 5 of the paper.
   (c) Did you discuss any potential negative societal impacts of your work? [N/A] No foreseeable negative societal impact.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes] More detailed proofs are in the supplemental material.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] The code and the data can be found in the supplemental material. See README.md for more details.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Section 5 of the paper and the supplemental material.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] We estimated the running times analytically: See Section 5.2. of the paper.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See the supplemental material.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] By assets, we refer to the original AdaPart implementation and the benchmark instances.
   (b) Did you mention the license of the assets? [No] The license of the test instances is mentioned in Section 5.2. However, the Github repository of the original AdaPart implementation contains no license, but we have discussed running experiments with it with its authors.
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] We include our code in the supplemental material.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A] The original AdaPart implementation and the graph data are publicly available on the Internet.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]