Homework 1

Ross B. Alexander (rbalexan@stanford.edu)

Problem 1 - Linear algebra

Are the following statements true or false?

a) The inverse of a symmetric matrix is symmetric.

Let $A \in S^n$:

$A \stackrel{\triangle}{=} A^T$ — A is symmetric

$A^{-1} = (A^T)^{-1}$ — A is invertible

$A^{-1} = (A^{-1})^T$ — inverse and transpose commute

Thus, if the inverse of a symmetric matrix exists, it is symmetric.

$\boxed{\text{true}}$

b) All 2x2 orthogonal matrices have the form

$$\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \text{ OR } \begin{bmatrix} \cos\theta & \sin\theta \\ \sin\theta & -\cos\theta \end{bmatrix}.$$

Let $A \in \mathbb{R}^{2\times 2}$, if A is orthogonal, then $AA^T = A^TA = I \iff A^T = A^{-1}$

$$AA^T = \begin{bmatrix} a & b \\ c & d \end{bmatrix}\begin{bmatrix} a & c \\ b & d \end{bmatrix} = \begin{bmatrix} a^2+b^2 & ac+bd \\ ac+bd & c^2+d^2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$\iff a^2+b^2 = 1$, $c^2+d^2 = 1$, $ac+bd = 0$

satisfied only for:

$a = c_\theta,\ b = -s_\theta,\ c = s_\theta,\ d = c_\theta$ $a = c_\theta,\ b = s_\theta,\ c = s_\theta,\ d = -c_\theta$

$$A^TA = \begin{bmatrix} a & c \\ b & d \end{bmatrix}\begin{bmatrix} a & b \\ c & d \end{bmatrix} = \begin{bmatrix} a^2+c^2 & ab+cd \\ ab+cd & b^2+d^2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$\iff a^2+c^2 = 1$, $b^2+d^2 = 1$, $ab+cd = 0$

satisfied only for:

$a = c_\theta,\ b = -s_\theta,\ c = s_\theta,\ d = c_\theta$ $a = c_\theta,\ b = s_\theta,\ c = s_\theta,\ d = -c_\theta$

Thus, all orthogonal matrices can be represented in the given forms.

$\boxed{\text{true}}$

c) For the following A, A can be written as CC^T for some matrix C.

$$A = \begin{bmatrix} -8 & -1 & -6 \\ -3 & -5 & -7 \\ -4 & -9 & -2 \end{bmatrix}$$

For an arbitrary $C \in \mathbb{R}^{3\times3}$, we examine CC^T,

$$(CC^T)^T = (C^T)^T C^T = CC^T$$

Therefore, CC^T is symmetric.

Since A is not symmetric, there is no representation of $A = CC^T$.

$\boxed{\text{false}}$

Problem 2 - Divide-and-conquer matrix multiplication

For a matrix A, $A^2 = AA$ is the square of A.

a) Show that five multiplications are sufficient to compute the square of a 2×2 matrix $A = \begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix}$ where a_1, a_2, a_3, a_4 are scalars.

$$AA = \begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix}\begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix}$$
$$= \begin{bmatrix} a_1^2 + a_2a_3 & a_1a_2 + a_2a_4 \\ a_1a_3 + a_3a_4 & a_2a_3 + a_4^2 \end{bmatrix}$$
$$= \begin{bmatrix} a_1^2 + a_2a_3 & a_2(a_1+a_4) \\ a_3(a_1+a_4) & a_4^2 + a_2a_3 \end{bmatrix}$$

We can see that we only need to complete five multiplications. a_1^2, a_2a_3, $a_2(a_1+a_4)$, $a_3(a_1+a_4)$, a_4^2. ■

b) Generalize the formula in a) to a 2×2 block matrix $A = \begin{bmatrix} A_1 & A_2 \\ A_3 & A_4 \end{bmatrix}$ where A_1, A_2, A_3, A_4 are arbitrary matrices.

$$AA = \begin{bmatrix} A_1 & A_2 \\ A_3 & A_4 \end{bmatrix}\begin{bmatrix} A_1 & A_2 \\ A_3 & A_4 \end{bmatrix}$$
$$= \begin{bmatrix} A_1A_1 + A_2A_3 & A_1A_2 + A_2A_4 \\ A_3A_1 + A_4A_3 & A_3A_2 + A_4A_4 \end{bmatrix}$$
$$= \begin{bmatrix} A_1^2 + A_2A_3 & A_1A_2 + A_2A_4 \\ A_3A_1 + A_4A_3 & A_3A_2 + A_4^2 \end{bmatrix}$$

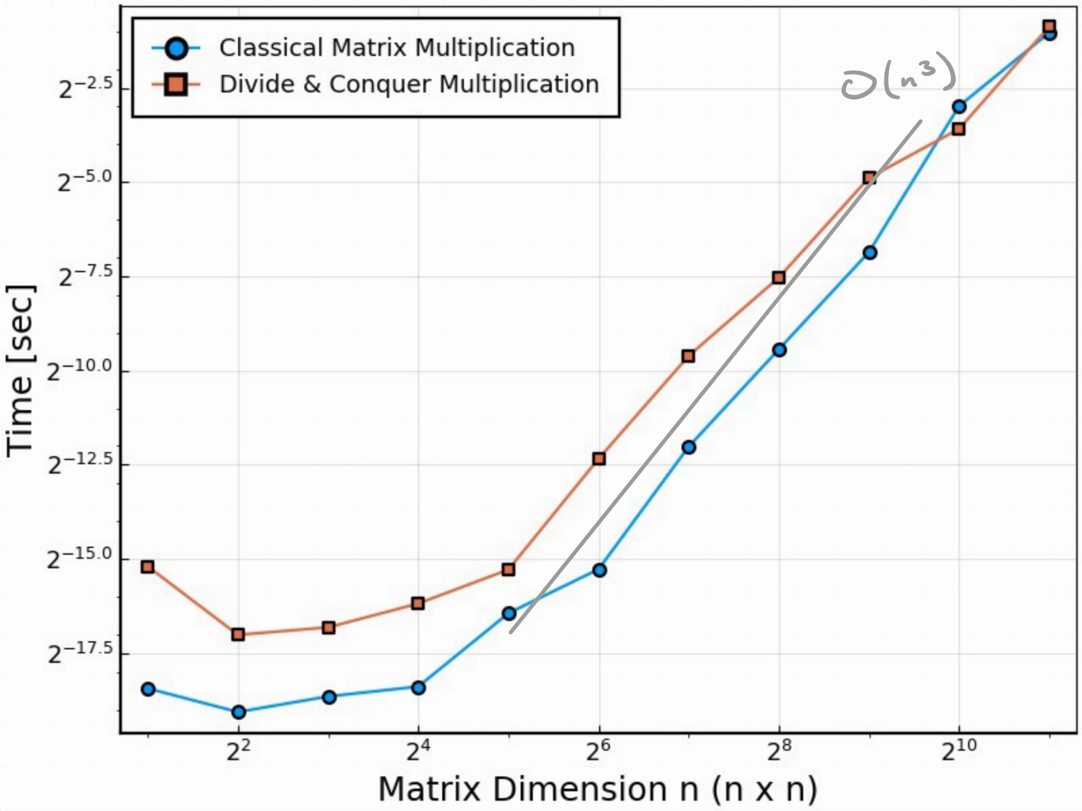
We can see that the generalized version requires more operations since the matrix products (previously scalar products) do not necessarily commute ($a_2a_3 = a_3a_2$, but not necessarily $A_2A_3 = A_3A_2$). Though we can efficiently compute A_1^2 & A_4^2 using 5 operations and we can use classical (or Strassen) matrix multiplication for the other products. We still save some operations regardless.

c) Instead of using the classical matrix multiplication (three-loop) algorithm for computing A^2, we may apply the block formula you derived in b) to reduce $2n \times 2n$ problems to several $n \times n$ computations, which can be tackled with classical matrix multiplication. Compare the total number of arithmetic operations. Generate $2n \times 2n$ random A matrices and plot the wall-clock time of the classical matrix multiplication algorithm and the algorithm you developed in b) for computing A^2 for $n = 4, \ldots, 10000$ (your system's memory limit). You can use standard packages for matrix multiplication e.g. numpy.matmul.

In classical matrix multiplication, A^2, $A \in \mathbb{R}^{n \times n}$ takes n^3 multiplications and n^3 additions.

In divide-and-conquer matrix multiplication, A^2, $A \in \mathbb{R}^{n \times n}$, we generate 8 $(n/2)$ classical matrix multiplication problems which take a total of $8 \cdot \left(\frac{n}{2}\right)^3 = n^3$ multiplications and $8 \cdot \left(\frac{n}{2}\right)^3 = n^3$ additions, plus 4 additions of $\left(\frac{n}{2}\right) \times \left(\frac{n}{2}\right)$ matrix blocks $= 4\left(\frac{n}{2}\right)^2 = n^2$ additions.

As a result, the complexity of classical matrix multiplication and divide-and-conquer multiplication is $\mathcal{O}(n^3)$.



d) Show that if you have an algorithm for squaring $n \times n$ matrices in $O(n^c)$ time, then you can use it to multiply any two arbitrary $n \times n$ matrices in $O(n^c)$ time. Hint: Consider AB; can you define a matrix whose square contains AB?

$$\boxed{\begin{array}{l}\text{If we can perform } AA \text{ in } O(n^c) \text{ time,} \\ \quad AB = (AB)^{\frac{1}{2}}(AB)^{\frac{1}{2}} \\ \quad AB = CC \\ \quad AB = C^2 \\ \text{That is, there is an equivalent representation of the product } AB \text{ as} \\ \text{the square of a matrix } C\text{, so if we can identify the matrix } C\text{,} \\ \text{we can compute } AB \text{ in } O(n^c).\end{array}}$$

Problem 3 - Probability

a) Random variables X and Y have a joint distribution $p(x,y)$. Prove the following results. You can assume continuous distributions for simplicity.

i) $\mathbb{E}[X] = \mathbb{E}_Y[\mathbb{E}_X[X|Y]]$

$$\mathbb{E}_Y[\mathbb{E}_X[X|Y]] = \int p(y)\left[\int \frac{p(x,y)}{p(y)} x \, dx\right] dy$$

$$= \int \frac{\cancel{p(y)}}{\cancel{p(y)}}\left[\int p(x,y) x \, dx\right] dy$$

$$\boxed{\mathbb{E}[X] = \iint p(x,y)\, x \, dx\, dy \quad \blacksquare}$$

ii) $\mathbb{E}[\mathbb{1}\{x \in C\}] = P(x \in C)$, where $\mathbb{1}\{x \in C\}$ is the indicator function

$$\mathbb{E}[\mathbb{1}\{x \in C\}] = \int p(x)\, \mathbb{1}\{x \in C\}\, dx$$

$$= \begin{cases} \int p(x)\,dx & x \in C \\ 0 & x \notin C \end{cases}$$

$$\boxed{\mathbb{E}[\mathbb{1}\{x \in C\}] = p(x \in C) \quad \blacksquare}$$

iii) $\mathrm{Var}[X] = \mathbb{E}_Y[\mathrm{Var}_X[X|Y] + \mathrm{Var}_Y[Y|X]]$

$$\mathbb{E}_Y[\mathrm{Var}_X[X|Y] + \mathrm{Var}_Y[Y|X]] = \mathbb{E}_Y[\mathrm{Var}_X[X|Y]] + \cancelto{0}{\mathbb{E}_Y[\mathrm{Var}_Y[Y|X]]}$$

$$= \int p(y) \int \frac{p(x,y)}{p(y)} (x - \mathbb{E}[X])^2 \, dx\, dy$$

$$= \int \frac{\cancel{p(y)}}{\cancel{p(y)}} \int p(x,y)(x - \mathbb{E}[X])^2 \, dx\, dy$$

$$\boxed{\mathrm{Var}[X] = \iint p(x,y)(x - \mathbb{E}[X])^2 \, dx\, dy \quad \blacksquare}$$

iv) if $X \perp Y$, then $\mathbb{E}[XY] = \mathbb{E}[X]\mathbb{E}[Y]$

- if $X \perp Y$, $p(x,y) = p(x)p(y)$

$$\mathbb{E}[XY] = \iint p(x,y)\, xy \, dx\, dy$$

$$= \iint p(x)p(y)\, xy \, dx\, dy$$

$$= \int p(x)\, x \, dx \int p(y)\, y \, dy$$

$$\boxed{\mathbb{E}[XY] = \mathbb{E}[X]\mathbb{E}[Y] \quad \blacksquare}$$

v) if $X, Y \in \{0,1\}$ and $\mathbb{E}[XY] = \mathbb{E}[X]\mathbb{E}[Y]$, then $X \perp Y$

$$\mathbb{E}[XY] = \mathbb{E}[X]\mathbb{E}[Y]$$

$$\sum_{x \in \{0,1\}} \sum_{y \in \{0,1\}} p(x,y)\, xy = \left[\sum_{x \in \{0,1\}} p(x)\, x\right]\left[\sum_{y \in \{0,1\}} p(y)\, y\right]$$

$$\boxed{p(x=1, y=1) = p(x=1)\, p(y=1) \quad \blacksquare}$$ (def'n. of independence)

b) Show that the approximate randomized counting algorithm described in Lemma 1 of Lecture 2 slides (p. 14) is unbiased: i.e. $\mathbb{E}[\tilde{n}] = n$.

$$\mathbb{E}[\tilde{n}] = \mathbb{E}[2^x - 1]$$
$$= \sum_{i=0}^{\infty} P(x=i)\, 2^x$$

- $P(X)$ depends on prior trials

$$P(x=i) = \frac{1}{2^x}(\ldots$$

$\boxed{\text{Not sure how to do this one}}$

c) Prove the variance formula in Lemma 2 of Lecture 2 slides (p. 38) for approximate matrix multiplication $AB \approx CR$

$$\mathrm{Var}[(CR)_{ij}] = \frac{1}{m}\sum_{k=1}^{d} \frac{A_{ik}^2 B_{kj}^2}{p_k} - \frac{1}{m}(AB)_{ij}^2$$

where $\{p_k\}_{k=1}^{d}$ are sampling probabilities.

$\boxed{\text{Ran out of time/energy. Will try later.}}$

Problem 4 – Positive semi-definite matrices

Let A be a real, symmetric $d \times d$ matrix, $A \in S^d$. We say A is positive semi-definite (PSD) if for all $x \in \mathbb{R}^d$, $x^T A x \geq 0$. We say A is positive definite (PD) if for all $x \in \mathbb{R}^d$, $x \neq 0$, $x^T A x > 0$. When A is PSD, we write $A \succeq 0$, when A is PD, we write $A \succ 0$.

The spectral theorem says that every real symmetric matrix A can be expressed $A = U \Lambda U^T$, where U is a $d \times d$ orthogonal matrix, such that $UU^T = U^T U = I$, and $\Lambda = \mathrm{diag}(\lambda_1, \dots, \lambda_d)$. Multiplying on the right by U, we see that $AU = U\Lambda$. Letting u_i be the ith column of U, we have $Au_i = \lambda_i u_i$ $\forall i$. This expression reveals that λ_i are eigenvalues of A and u_i are the eigenvectors associated with λ_i.

Prove the following statements

a) A is PSD iff $\lambda_i \geq 0$ $\forall i$

For an arbitrary $A \in S^n$, $A = U\Lambda U^T$, A is PSD if

$$\begin{aligned} 0 &\leq x^T A x \quad \forall x \\ &\leq x^T (U \Lambda U^T) x \\ &\leq (x^T U) \Lambda (U^T x) \\ &\leq (U^T x)^T \Lambda (U^T x) \qquad \text{let } U^T x = \tilde{x} \\ &\leq \tilde{x}^T \Lambda \tilde{x} \end{aligned}$$

$$\boxed{0 \leq \sum_{i=1}^{n} \tilde{x}_i^2 \lambda_i \iff \lambda_i \geq 0 \quad \blacksquare}$$

b) A is PD iff $\lambda_i > 0$ $\forall i$

For an arbitrary $A \in S^n$, $A = U\Lambda U^T$, A is PD if

$$\begin{aligned} 0 &< x^T A x \quad \forall x \neq 0 \\ &< x^T (U \Lambda U^T) x \\ &< (x^T U) \Lambda (U^T x) \\ &< (U^T x)^T \Lambda (U^T x) \qquad \text{let } U^T x = \tilde{x} \\ &< \tilde{x}^T \Lambda \tilde{x} \end{aligned}$$

$$\boxed{0 < \sum_{i=1}^{n} \tilde{x}_i^2 \lambda_i \iff \lambda_i > 0 \quad \blacksquare}$$

Problem 5 – Norms

a) For $p = 1, 2, \infty$, verify that $\|\cdot\|_p$ are norms.

- $p = 1 \longrightarrow \|x\|_1 = \sum_{i=1}^{n} |x_i|$
  - positive $\longrightarrow$ trivial to see that $\|x\|_1 \geq 0 \;\forall x$, and $\|x\|_1 = 0$ iff $x = 0$ ✓
  - positively homogeneous

$$\|\alpha x\|_1 = |\alpha| \|x\|_1$$
$$\sum_{i=1}^{n} |\alpha x_i| = |\alpha| \sum_{i=1}^{n} |x_i|$$
$$\sum_{i=1}^{n} |\alpha| |x_i| = |\alpha| \sum_{i=1}^{n} |x_i|$$
$$|\alpha| \sum_{i=1}^{n} |x_i| = |\alpha| \sum_{i=1}^{n} |x_i| \quad ✓$$

  - triangle inequality

$$\|x+y\|_1 \leq \|x\|_1 + \|y\|_1$$
$$\sum_{i=1}^{n} |x_i + y_i| \leq \sum_{i=1}^{n} |x_i| + \sum_{i=1}^{n} |y_i|$$

  - sufficient to consider $n = 1$

$$|x+y| \leq |x| + |y| \quad ✓$$ (triangle inequality for absolute value)

$$\boxed{\text{Therefore, the } \ell_1 \text{ function } \|\cdot\|_1 \text{ is a norm on } \mathbb{R}^n. \blacksquare}$$

- $p = 2 \longrightarrow \|x\|_2 = \left(\sum_{i=1}^{n} x_i^2\right)^{\frac{1}{2}}$
  - positive $\longrightarrow$ trivial to see that $\|x\|_2 \geq 0 \;\forall x$, and $\|x\|_2 = 0$ iff $x = 0$ ✓
  - positively homogeneous

$$\|\alpha x\|_2 = |\alpha| \|x\|_2$$
$$\left(\sum_{i=1}^{n} (\alpha x_i)^2\right)^{\frac{1}{2}} = |\alpha| \left(\sum_{i=1}^{n} x_i^2\right)^{\frac{1}{2}}$$
$$\left(\sum_{i=1}^{n} \alpha^2 x_i^2\right)^{\frac{1}{2}} = |\alpha| \left(\sum_{i=1}^{n} x_i^2\right)^{\frac{1}{2}}$$
$$|\alpha| \left(\sum_{i=1}^{n} x_i^2\right)^{\frac{1}{2}} = |\alpha| \left(\sum_{i=1}^{n} x_i^2\right)^{\frac{1}{2}} \quad ✓$$

  - triangle inequality

$$\|x+y\|_2 \leq \|x\|_2 + \|y\|_2$$
$$\left(\sum_{i=1}^{n} (x_i + y_i)^2\right)^{\frac{1}{2}} \leq \left(\sum_{i=1}^{n} x_i^2\right)^{\frac{1}{2}} + \left(\sum_{i=1}^{n} y_i^2\right)^{\frac{1}{2}}$$
$$\sum_{i=1}^{n} (x_i + y_i)^2 \leq \sum_{i=1}^{n} x_i^2 + \sum_{i=1}^{n} y_i^2 + 2\left(\sum_{i=1}^{n} x_i^2\right)^{\frac{1}{2}} \left(\sum_{i=1}^{n} y_i^2\right)^{\frac{1}{2}}$$
$$\cancel{\sum_{i=1}^{n} x_i^2} + \cancel{\sum_{i=1}^{n} y_i^2} + 2\sum_{i=1}^{n} x_i y_i \leq \cancel{\sum_{i=1}^{n} x_i^2} + \cancel{\sum_{i=1}^{n} y_i^2} + 2\left(\sum_{i=1}^{n} x_i^2\right)^{\frac{1}{2}} \left(\sum_{i=1}^{n} y_i^2\right)^{\frac{1}{2}}$$
$$\sum_{i=1}^{n} x_i y_i \leq \left(\sum_{i=1}^{n} x_i^2\right)^{\frac{1}{2}} \left(\sum_{i=1}^{n} y_i^2\right)^{\frac{1}{2}}$$
$$\langle x, y \rangle \leq \|x\|_2 \|y\|_2 \quad ✓$$ (Cauchy-Schwarz inequality)

$$\boxed{\text{Therefore, the } \ell_2 \text{ function } \|\cdot\|_2 \text{ is a norm on } \mathbb{R}^n. \blacksquare}$$

- $p=\infty \to \|x\|_\infty = \max_i |x_i|$
  - positive $\to$ trivial to see that $\|x\|_\infty \geq 0 \ \forall x$, and $\|x\|_\infty = 0$ iff $x=0$ ✓
  - positively homogeneous
    $\|\alpha x\|_\infty = |\alpha| \|x\|_\infty$
    $\max_i |\alpha x_i| = |\alpha| \max_i |x_i|$
    $|\alpha| \max_i |x_i| = |\alpha| \max_i |x_i|$ ✓
  - triangle inequality
    $\|x+y\|_\infty \leq \|x\|_\infty + \|y\|_\infty$
    $\max_i |x_i+y_i| \leq \max_i |x_i| + \max_i |y_i|$ ✓ (guaranteed)

$\boxed{\text{Therefore, the } l_\infty \text{ function } \|\cdot\|_\infty \text{ is a norm on } \mathbb{R}^n. \ \blacksquare}$

Then, for a vector $x \in \mathbb{R}^n$, show that

$$\|x\|_\infty \leq \|x\|_2 \leq \|x\|_1 \leq \sqrt{n}\,\|x\|_2 \leq n\|x\|_\infty$$

and provide an example demonstrating that each inequality can be exact.

We have for $x = \begin{bmatrix} 3 \\ 4 \end{bmatrix}$:

$$\begin{array}{ccccccccc} \|x\|_\infty & \leq & \|x\|_2 & \leq & \|x\|_1 & \leq & \sqrt{n}\,\|x\|_2 & \leq & n\|x\|_\infty \\ 4 & \leq & 5 & \leq & 7 & \leq & \sqrt{2}(5) & \leq & 2(4) \\ \sqrt{16} & \leq & \sqrt{25} & \leq & \sqrt{49} & \leq & \sqrt{50} & \leq & \sqrt{64} \end{array} \ \blacksquare$$

The inequalities can be exact if $n=1$, e.g. for $x=[5]$

$$\begin{array}{ccccccccc} \|x\|_\infty & \leq & \|x\|_2 & \leq & \|x\|_1 & \leq & \sqrt{n}\,\|x\|_2 & \leq & n\|x\|_\infty \\ 5 & \leq & 5 & \leq & 5 & \leq & \sqrt{1}(5) & \leq & 1(5) \\ 5 & = & 5 & = & 5 & = & 5 & = & 5 \end{array} \ \blacksquare$$

b) For vectors $x, y \in \mathbb{R}^n$, show that $|x^T y| \leq \|x\|_2 \|y\|_2$ with equality if and only if x and y are linearly dependent. More generally, show that $x^T y \leq \|x\|_1 \|y\|_\infty$. Note that this implies $\|x\|_2^2 \leq \|x\|_1 \|x\|_\infty$; and that these are special cases of Hölder's inequality.

$$|x^T y| \leq \|x\|_2 \|y\|_2$$
$$\left|\sum_{i=1}^n x_i y_i\right| \leq \left(\sum_{i=1}^n x_i^2\right)^{\frac{1}{2}} \left(\sum_{i=1}^n y_i^2\right)^{\frac{1}{2}}$$
$$\left(\sum_{i=1}^n x_i y_i\right)^2 \leq \left(\sum_{i=1}^n x_i^2\right)\left(\sum_{i=1}^n y_i^2\right)$$

- if x and y are linearly dependent, then $y = \alpha x$ for some $\alpha \in \mathbb{R}$, $\alpha \neq 0$

$$\left(\sum_{i=1}^n x_i(\alpha x_i)\right)^2 \leq \left(\sum_{i=1}^n x_i^2\right)\left(\sum_{i=1}^n (\alpha x_i)^2\right)$$
$$\left(\alpha \sum_{i=1}^n x_i^2\right)^2 \leq \left(\sum_{i=1}^n x_i^2\right)\left(\alpha^2 \sum_{i=1}^n x_i^2\right)$$

$$\boxed{\alpha^2\left(\sum_{i=1}^n x_i^2\right)^2 = \alpha^2\left(\sum_{i=1}^n x_i^2\right)^2 \ \blacksquare}$$

Showing that

$$x^T y \le \|x\|_1 \|y\|_\infty$$

$$\sum_{i=1}^{n} x_i y_i \le \left(\sum_{i=1}^{n} |x_i|\right) \max_i |y_i|$$

$$\boxed{\sum_{i=1}^{n} x_i y_i \le \sum_{i=1}^{n} |x_i| \max_j |y_j| \quad \blacksquare}$$

straightforward to see that LHS always $\le$ RHS

c) For $A \in \mathbb{R}^{m\times n}$, show that $\mathrm{tr}(A^T A) = \sum_{i=1}^{m}\sum_{j=1}^{n} A_{ij}^2$, and show that $\sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} A_{ij}^2}$ is a norm on $m\times n$ matrices. This is the Frobenius norm $\|\cdot\|_F$. Show that, in addition to satisfying the defining properties of a norm, the Frobenius norm is a submultiplicative norm, i.e.

$$\|AB\|_F \le \|A\|_F \|B\|_F \quad \text{whenever } AB \text{ is defined.}$$

We have:

$$\mathrm{tr}(A^T A) = \sum_{i=1}^{m}\sum_{j=1}^{n} A_{ij}^2$$

$$\mathrm{tr}(A^T A) = \sum_{i=1}^{n} (A^T A)_{ii}$$

$$= \sum_{i=1}^{n}\sum_{k=1}^{m} (A_{kj} A_{kl})_{ii} \qquad \Rightarrow ii \text{ when } j=l$$

$$= \sum_{i=1}^{n}\sum_{k=1}^{m} A_{ki} A_{ki}$$

$$\boxed{\mathrm{tr}(A^T A) = \sum_{i=1}^{n}\sum_{j=1}^{n} A_{ij}^2 \quad \blacksquare}$$

Proving that our Frobenius function $\|\cdot\|_F = \left(\sum_{i=1}^{m}\sum_{j=1}^{n} A_{ij}^2\right)^{\frac{1}{2}}$ is a norm on $\mathbb{R}^{m\times n}$, we have

- positive $\longrightarrow$ trivial to see that $\|A\|_F \ge 0 \ \forall A$, and $\|A\|_F = 0$ iff $A = 0$ ✓
- positively homogeneous

$$\|\alpha A\|_F = |\alpha| \|A\|_F$$

$$\left(\sum_{i=1}^{m}\sum_{j=1}^{n} (\alpha A_{ij})^2\right)^{\frac{1}{2}} = |\alpha| \left(\sum_{i=1}^{m}\sum_{j=1}^{n} A_{ij}^2\right)^{\frac{1}{2}}$$

$$|\alpha| \left(\sum_{i=1}^{m}\sum_{j=1}^{n} A_{ij}^2\right)^{\frac{1}{2}} = |\alpha| \left(\sum_{i=1}^{m}\sum_{j=1}^{n} A_{ij}^2\right)^{\frac{1}{2}} \ ✓$$

- triangle inequality

$$\|A+B\|_F \le \|A\|_F + \|B\|_F$$

$$\left(\sum_{i=1}^{m}\sum_{j=1}^{n} (A_{ij}+B_{ij})^2\right)^{\frac{1}{2}} \le \left(\sum_{i=1}^{m}\sum_{j=1}^{n} A_{ij}^2\right)^{\frac{1}{2}} + \left(\sum_{i=1}^{m}\sum_{j=1}^{n} B_{ij}^2\right)^{\frac{1}{2}}$$

$$\sum_{i=1}^{m}\sum_{j=1}^{n} (A_{ij}+B_{ij})^2 \le \sum_{i=1}^{m}\sum_{j=1}^{n} A_{ij}^2 + \sum_{i=1}^{m}\sum_{j=1}^{n} B_{ij}^2 + 2\left(\sum_{i=1}^{m}\sum_{j=1}^{n} A_{ij}^2\right)^{\frac{1}{2}}\left(\sum_{i=1}^{m}\sum_{j=1}^{n} B_{ij}^2\right)^{\frac{1}{2}}$$

$$\cancel{\sum_{i=1}^{m}\sum_{j=1}^{n} A_{ij}^2} + \cancel{\sum_{i=1}^{m}\sum_{j=1}^{n} B_{ij}^2} + 2\sum_{i=1}^{m}\sum_{j=1}^{n} A_{ij}B_{ij} \le \cancel{\sum_{i=1}^{m}\sum_{j=1}^{n} A_{ij}^2} + \cancel{\sum_{i=1}^{m}\sum_{j=1}^{n} B_{ij}^2} + 2\left(\sum_{i=1}^{m}\sum_{j=1}^{n} A_{ij}^2\right)^{\frac{1}{2}}\left(\sum_{i=1}^{m}\sum_{j=1}^{n} B_{ij}^2\right)^{\frac{1}{2}}$$

$$\sum_{i=1}^{m}\sum_{j=1}^{n} A_{ij}B_{ij} \le \left(\sum_{i=1}^{m}\sum_{j=1}^{n} A_{ij}^2\right)^{\frac{1}{2}}\left(\sum_{i=1}^{m}\sum_{j=1}^{n} B_{ij}^2\right)^{\frac{1}{2}}$$

$$\langle A, B\rangle \le \|A\|_F \|B\|_F \ ✓$$

(Cauchy-Schwarz inequality)

$$\boxed{\text{Therefore the Frobenius function } \|\cdot\|_F \text{ is a norm on } \mathbb{R}^{m\times n}.}$$

Proving the Frobenius norm's submultiplicativity

$$\|AB\|_F = \left(\sum_{i=1}^{m}\sum_{j=1}^{n}\left(\sum_{k=1}^{d} A_{ik}B_{kj}\right)^2\right)^{\frac{1}{2}}$$

$$\|AB\|_F^2 = \sum_{i=1}^{m}\sum_{j=1}^{n}\left(\sum_{k=1}^{d} A_{ik}B_{kj}\right)^2$$

$$\leq \left(\sum_{i=1}^{m}\sum_{j=1}^{n} A_{ij}^2\right)\left(\sum_{i=1}^{m}\sum_{j=1}^{n} B_{ij}^2\right)$$

$$\|AB\|_F^2 \leq \|A\|_F^2 \|B\|_F^2$$

$$\boxed{\|AB\|_F \leq \|A\|_F \|B\|_F \quad \blacksquare}$$

d) Recall the definition of the spectral norm of an $m \times n$ matrix A:
$\|A\|_2 = \sqrt{\lambda_{max}(A^TA)} = \sigma_{max}(A)$, where $\lambda_{max}(A^TA)$ is the largest eigenvalue of A^TA and $\sigma_{max}(A)$ is the largest singular value of A.
Show that the Frobenius norm and the spectral norm are unitarily invariant, i.e. for U, V unitary (orthogonal in the real case), then $\|U^TAV\|_\xi = \|A\|_\xi$, for $\xi = 2, F$.

- for $\xi = 2 \rightarrow \|A\|_2 = \sqrt{\lambda_{max}(A^TA)} = \sigma_{max}(A)$
  - showing unitary invariance

$$\|A\|_2 = \|U^TAV\|_2$$

$$(\lambda_{max}(A^TA))^{\frac{1}{2}} = (\lambda_{max}(V^TA^T\cancel{UU^T}AV))^{\frac{1}{2}}$$

$$= (\lambda_{max}(V^TA^TAV))^{\frac{1}{2}}$$

$$= (\lambda_{max}(V^T(A^TA)V))^{\frac{1}{2}}$$

$\Downarrow$ spectral decomposition of $A^TA = \hat{U}^T\Lambda\hat{U}$

$$(\lambda_{max}(\hat{U}^T\Lambda\hat{U}))^{\frac{1}{2}} = (\lambda_{max}(V^T\hat{U}^T\Lambda\hat{U}V))^{\frac{1}{2}} \qquad \text{let } \hat{U}V = \underset{\sim}{U}$$

$$\boxed{(\lambda_{max}(\hat{U}^T\Lambda\hat{U}))^{\frac{1}{2}} = (\lambda_{max}(\underset{\sim}{U}^T\Lambda\underset{\sim}{U}))^{\frac{1}{2}} \quad \blacksquare}$$

- for $\xi = F \rightarrow \|A\|_F = \left(\sum_{i=1}^{m}\sum_{j=1}^{n} A_{ij}^2\right)^{\frac{1}{2}} = tr(A^TA)^{\frac{1}{2}}$
  - showing unitary invariance

$$\|A\|_F = \|U^TAV\|_F$$

$$tr(A^TA)^{\frac{1}{2}} = tr(V^TA^TUU^TAV)^{\frac{1}{2}}$$

$$tr(A^TA)^{\frac{1}{2}} = tr(A^T\cancel{UU^T}A\cancel{VV^T})^{\frac{1}{2}}$$

$$\boxed{tr(A^TA)^{\frac{1}{2}} = tr(A^TA)^{\frac{1}{2}} \quad \blacksquare}$$

Problem 6 - Approximate matrix multiplication

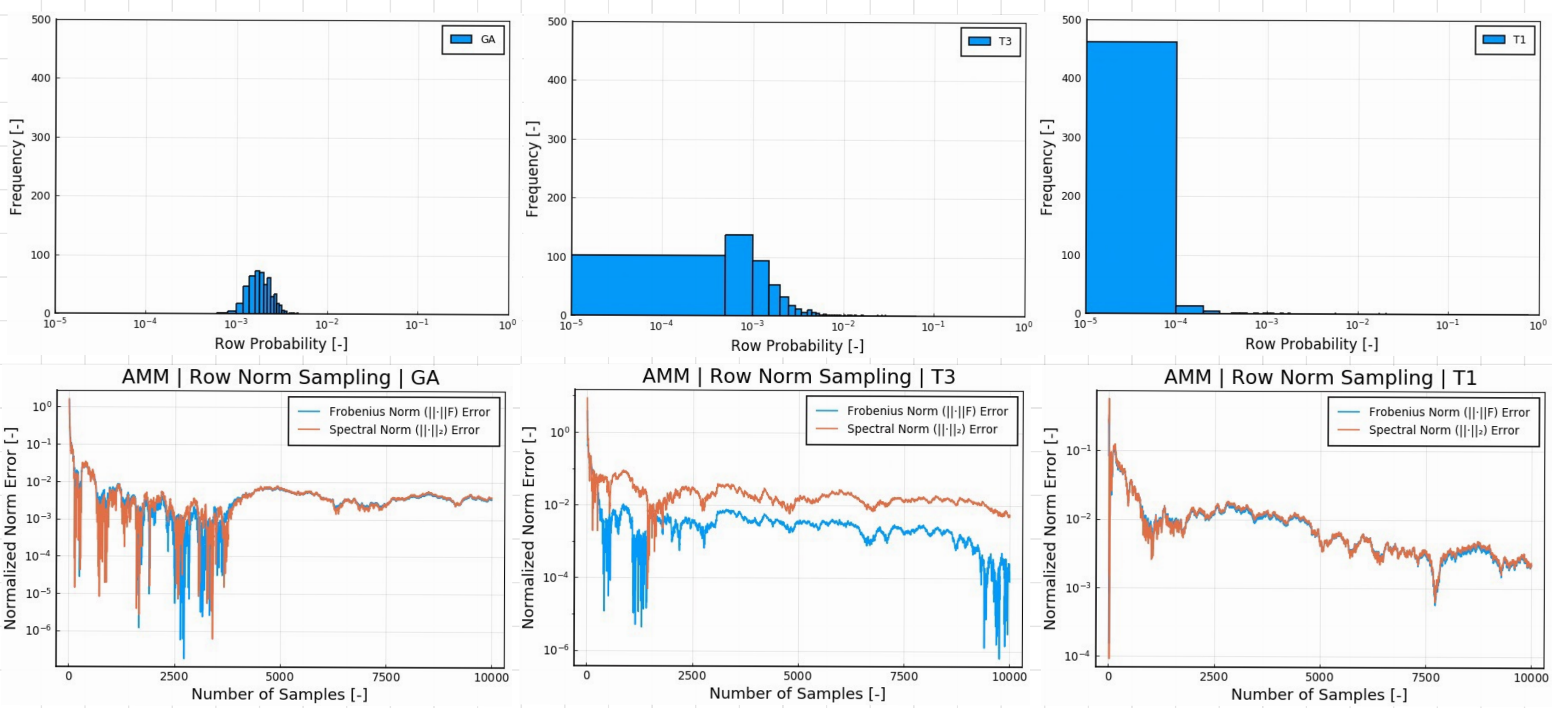
Here, we will consider the empirical performance of random sampling and random projection algorithms for approximating the product of two matrices. You may use any software package you prefer. Please be sure to describe what you used in sufficient detail so that someone else could reproduce your results. Let A be an $n \times d$ matrix with $n \gg d$ and consider approximating $A^T A$. First, generate the matrices A from one of three different classes of distributions described below.

GA data: $A \sim \mathcal{N}(1_d, \Sigma)$ where $\Sigma_{ij} = 2 \cdot 0.5^{|i-j|}$
T3 data: $A \sim t(\nu = 3, \Sigma)$ where $\Sigma_{ij} = 2 \cdot 0.5^{|i-j|}$
T1 data: $A \sim t(\nu = 1, \Sigma)$ where $\Sigma_{ij} = 2 \cdot 0.5^{|i-j|}$

To start, consider matrices of size $n \times d = 500 \times 50$. Generate three matrices, one from each method.

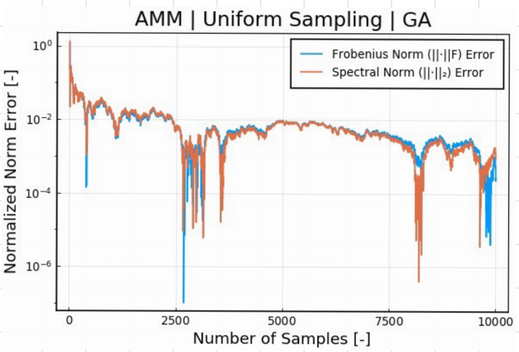
a) For each matrix A, approximate $A^T A$ with the random sampling algorithm we discussed in class, i.e. by sampling with respect to a probability distribution that depends on the norm-squared of the rows of the input matrix. Plot the probability distribution. Does it look uniform or nonuniform? Plot the performance of the spectral and Frobenius norm error as a function of the number of samples.

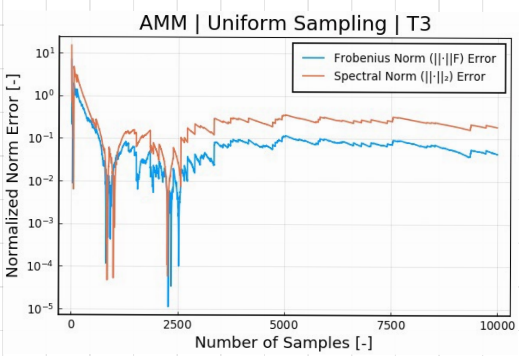
$$\boxed{\text{For GA data, the row probabilities are relatively uniform, while for the T3 and T1 data, there are more outlier row probabilities. See attached plots and code.}}$$

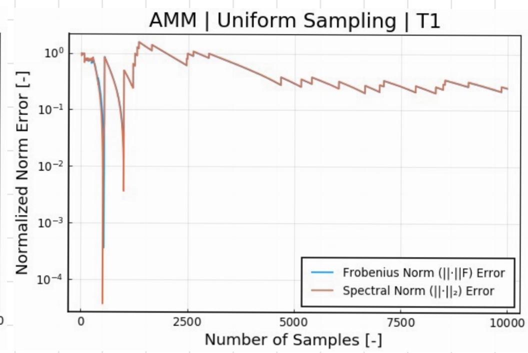


b) For each matrix A, approximate A^TA with the random sampling algorithm we discussed in class, except now use a uniform distribution instead of the norm-squared distribution to produce samples. Plot the performance of the spectral and Frobenius norm error as a function of the number of samples. For which matrices are the results similar and for which are they different relative to a)?

The GA data leads to similar convergence, while the T3 and T1 data lead to poorer results, likely since important rows are sampled infrequently – this leads to many of the samples having very low impact (periods of no/slow progress).

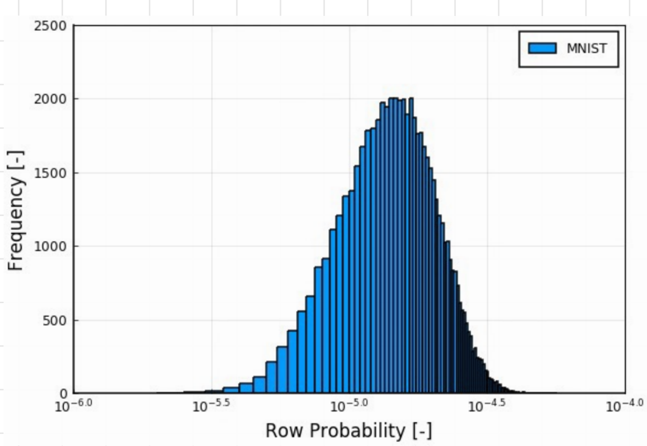


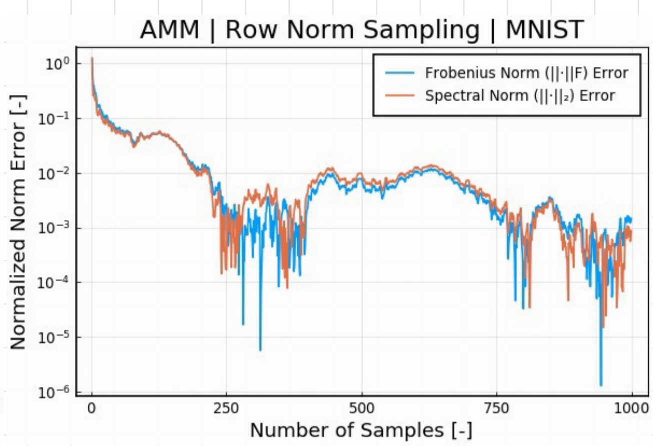




c) Now you will implement the matrix approximation technique on the MNIST dataset for handwritten digit classification. In the .mat file, you will find a matrix $A \in \mathbb{R}^{60000 \times 784}$. For this matrix, approximate A^TA with the random sampling algorithm we discussed in class, i.e. by sampling with respect to a probability distribution that depends on the norm-squared of the rows of the input matrix. Plot the probability distribution. Does it look uniform or nonuniform? Plot the performance of the spectral and Frobenius norm error as a function of the number of samples.

The distribution of row probabilities is relatively uniform. We observe good convergence of $\|A^TA\|_F$ and $\|A^TA\|_2$.





SEE ATTACHED CODE

In [1]:
```
using BenchmarkTools
using Plots; pyplot()
```

Out[1]:
```
Plots.PyPlotBackend()
```

Divide-and-conquer matrix multiplication

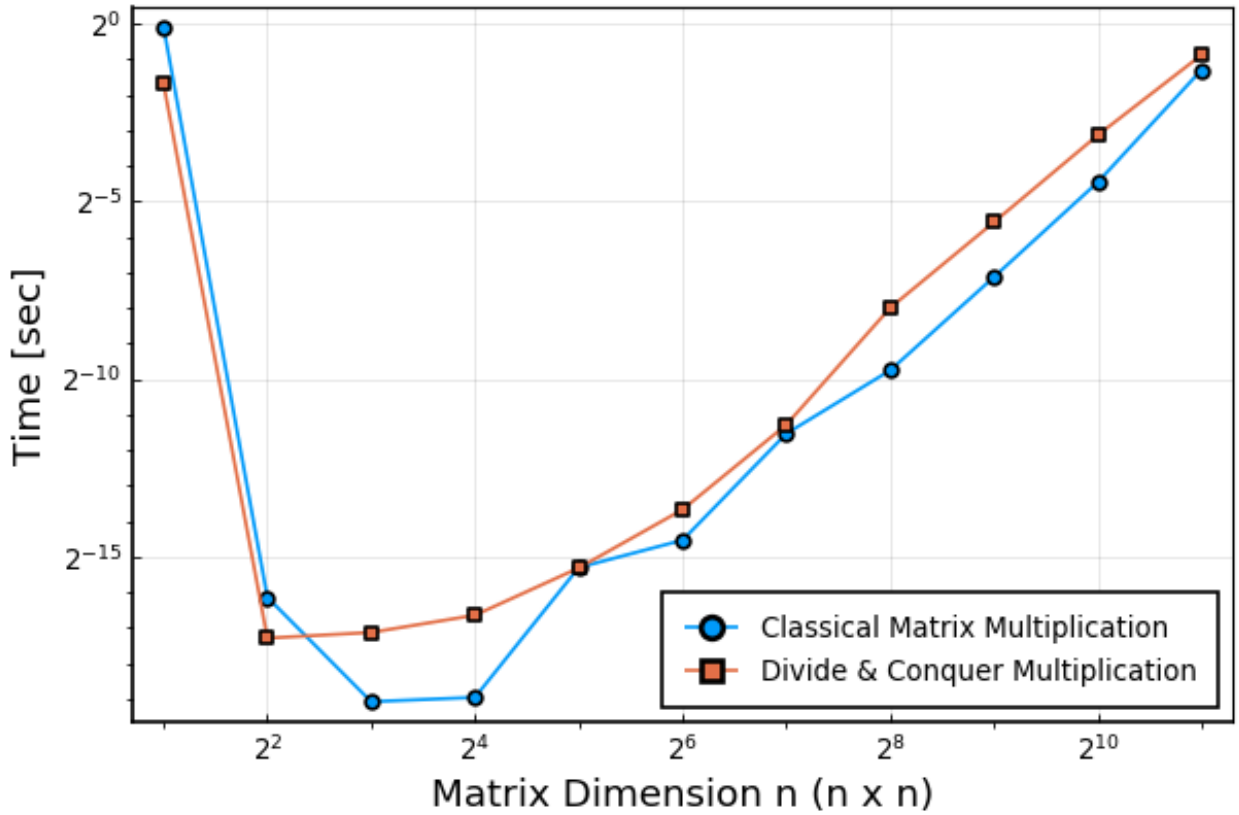
In [2]:
```
function divide_and_conquer(A, B)

    n = size(A)[1]

    if n == 1
        return A*B
    end

    n2 = Int(n/2)

    prod = zeros(n, n)

    prod[1:n2,     1:n2]      = A[1:n2,      1:n2]*B[1:n2, 1:n2]      + A[1:n2,      n2+1:end]*B[n2+1:end, 1:n2]
    prod[1:n2,     n2+1:end] = A[1:n2,      1:n2]*B[1:n2, n2+1:end] + A[1:n2,      n2+1:end]*B[n2+1:end, n2+1:end]
    prod[n2+1:end, 1:n2]      = A[n2+1:end, 1:n2]*B[1:n2, 1:n2]      + A[n2+1:end, n2+1:end]*B[n2+1:end, 1:n2]
    prod[n2+1:end, n2+1:end] = A[n2+1:end, 1:n2]*B[1:n2, n2+1:end] + A[n2+1:end, n2+1:end]*B[n2+1:end, n2+1:end]

    return prod

end
```

Out[2]:
```
divide_and_conquer (generic function with 1 method)
```

In [3]:
```
ns = Int.(2 .^ range(1, 11, step=1))
time_cmm = zeros(11)
time_dcm = zeros(11)

for (i, n) in enumerate(ns)

    A = rand(n, n)

    val, time_cmm[i] = @timed A*A
    val, time_dcm[i] = @timed divide_and_conquer(A, A)

end
```

In [4]:
```
plot( ns, time_cmm, xscale=:log2, yscale=:log2, label="Classical Matri
x Multiplication",
        box=:on, thickness_scaling=1.2, marker=:auto)#, size=(800,600)
)
plot!(ns, time_dcm, label="Divide & Conquer Multiplication", marker=:a
uto)
xlabel!("Matrix Dimension n (n x n)")
ylabel!("Time [sec]")
#png("cmm_v_dcm.png")
```

Out[4]:



AMM on sampled data (multivariate Gaussian, multivariate t-distribution)

In [5]:
```
using LinearAlgebra
using Distributions
```

In [6]:
```
n = 500
d = 50;
```

In [7]:
```
μ = ones(d)
Σ = [2*0.5^abs(i-j) for i in 1:d, j in 1:d]

μt = zeros(d)

GA = MvNormal(μ, Σ)
T1 = MvTDist(1, μt, Σ)
T3 = MvTDist(3, μt, Σ)

function rand_matrix_from_row_dist(dist, n, d)

    A = zeros(n, d)

    for i in 1:n
        A[i, :] = rand(dist)
    end

    return A

end
```

Out[7]: `rand_matrix_from_row_dist (generic function with 1 method)`

In [8]:
```
A_GA = rand_matrix_from_row_dist(GA, n, d)
A_T3 = rand_matrix_from_row_dist(T3, n, d)
A_T1 = rand_matrix_from_row_dist(T1, n, d);
```

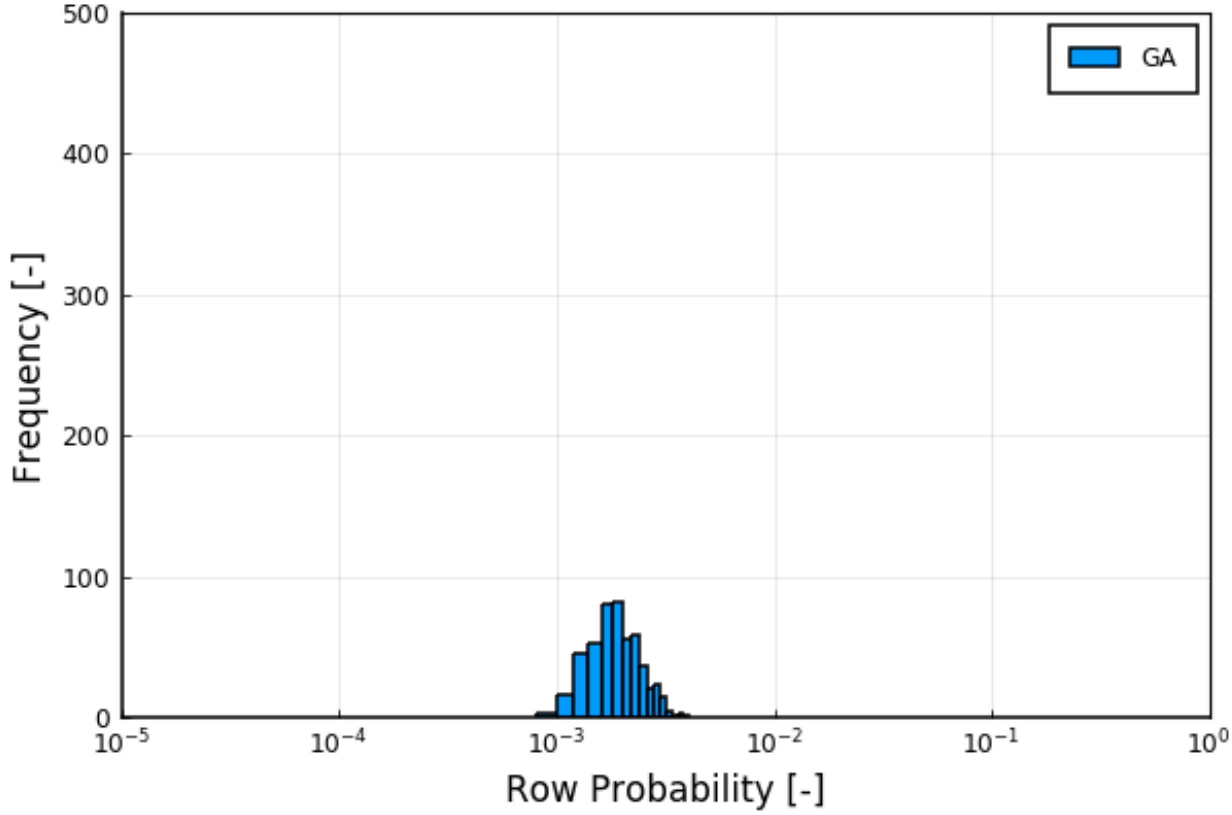
In [9]:
```
function row_score_probabilities(A)

    row_scores = [norm(row, 2)^2 for row in eachrow(A)]
    probs      = normalize(row_scores, 1)

end
```

Out[9]: `row_score_probabilities (generic function with 1 method)`

In [10]:
```
p_GA = row_score_probabilities(A_GA)
p_T3 = row_score_probabilities(A_T3)
p_T1 = row_score_probabilities(A_T1);
```

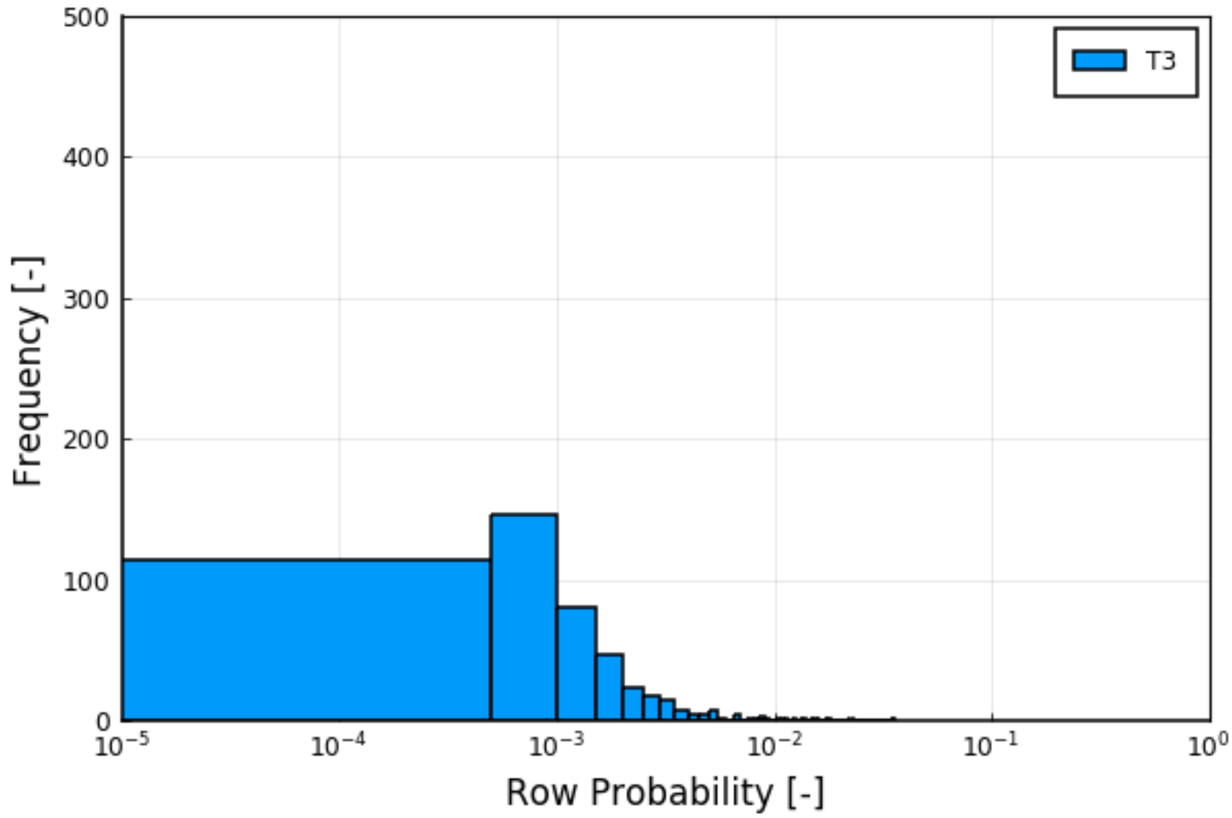
In [11]:
```
histogram(p_GA, label="GA",
            xscale=:log10, xlims=(1E-5, 1), ylims=(0, 500), thickness_
scaling=1.1, box=:on,
            xlabel="Row Probability [-]", ylabel="Frequency [-]")
#png("row_score_dist_ga.png")
```

Out[11]:



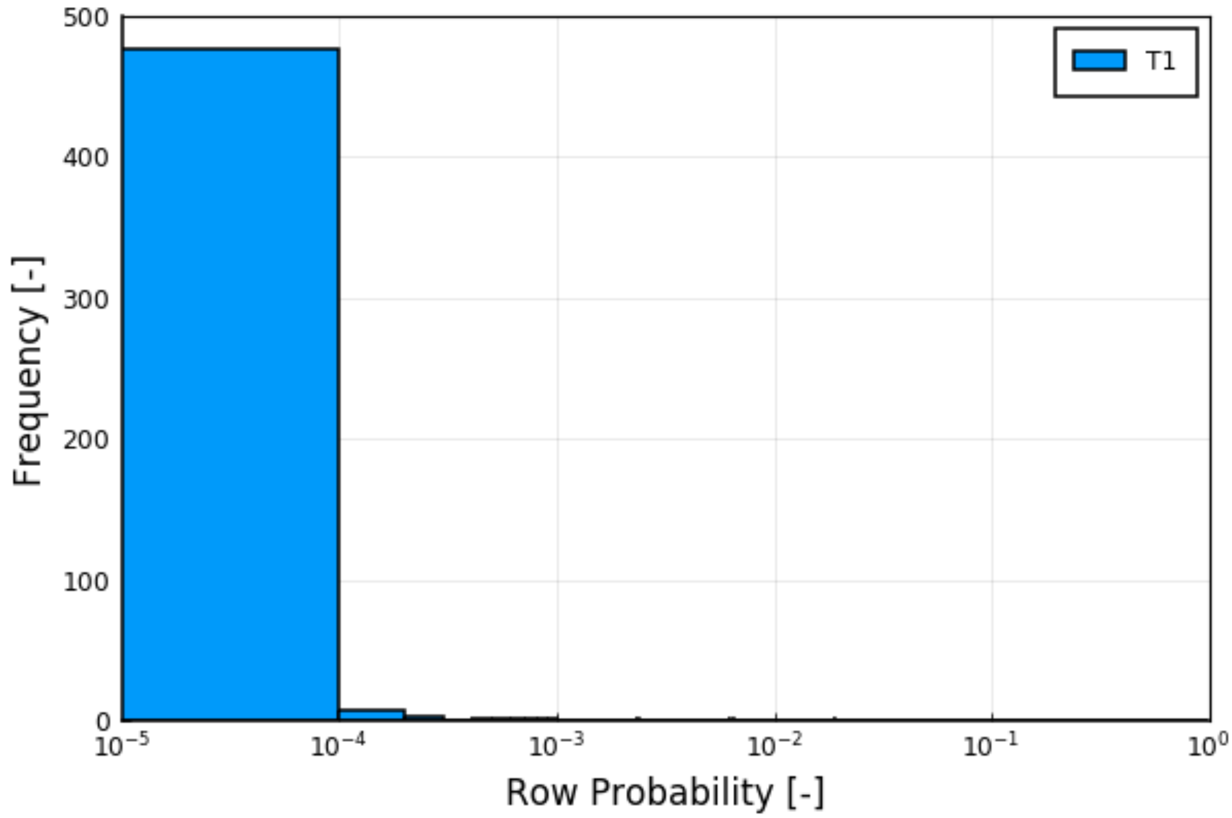
In [12]:
```
histogram(p_T3, label="T3",
            xscale=:log10, xlims=(1E-5, 1), ylims=(0, 500), thickness_
scaling=1.1, box=:on,
            xlabel="Row Probability [-]", ylabel="Frequency [-]")
#png("row_score_dist_t3.png")
```

Out[12]:



In [13]:
```
histogram(p_T1, label="T1",
            xscale=:log10, xlims=(1E-5, 1), ylims=(0, 500), thickness_
scaling=1.1, box=:on,
            xlabel="Row Probability [-]", ylabel="Frequency [-]")
#png("row_score_dist_t1.png")
```

Out[13]:



In [14]:
```julia
function approximate_matrix_multiplication(A, B, dist, m)

    p = probs(dist)

    AB = zeros(size(A)[1], size(B)[2], )

    AB_exact    = A * B
    frob_exact =   norm(AB_exact)
    spec_exact = opnorm(AB_exact)

    frob_error = zeros(m)
    spec_error = zeros(m)

    for i in 1:m

        ik = rand(dist)
        pk = p[ik]

        AB *= i-1
        AB += 1/pk * A[:, ik] * B[ik, :]'
        AB /= i

        frob_error[i] = abs(  norm(AB) - frob_exact) / frob_exact
        spec_error[i] = abs(opnorm(AB) - spec_exact) / spec_exact

    end

    return prod, frob_error, spec_error

end
```

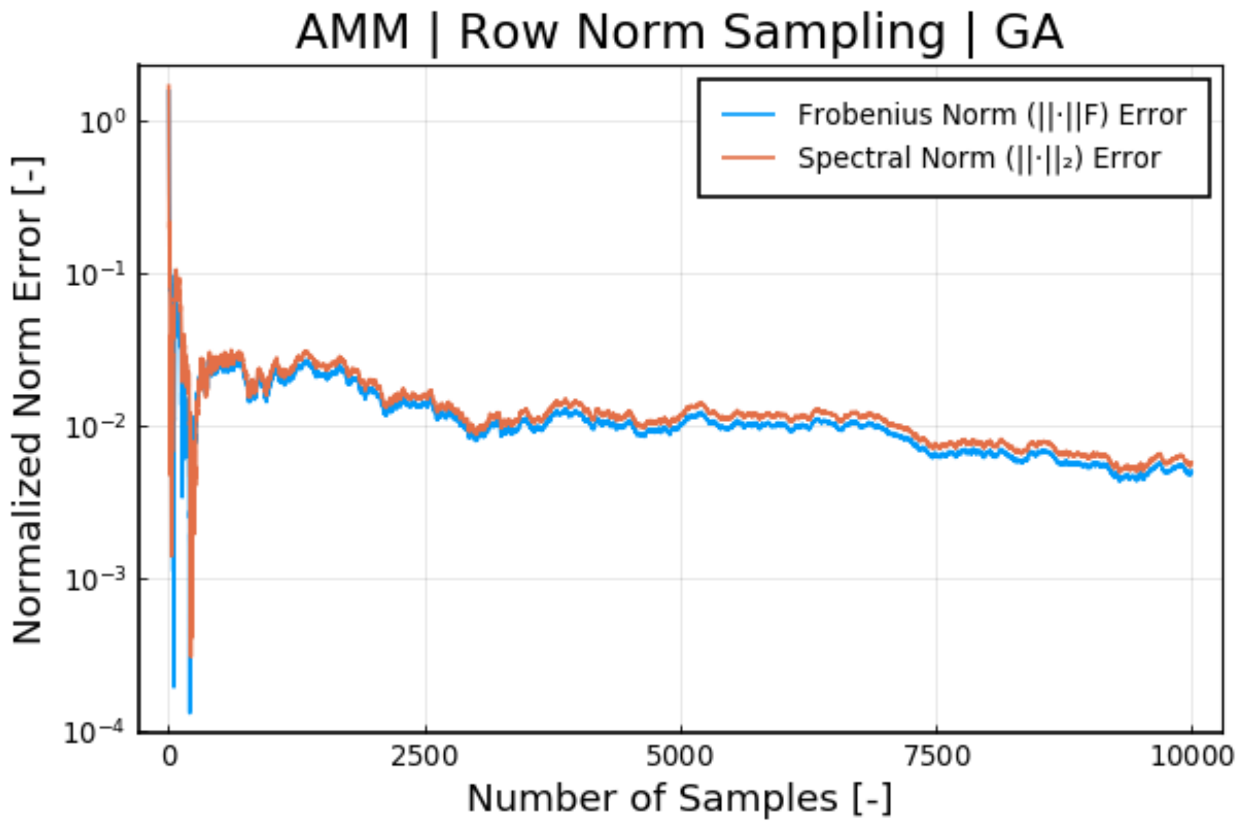
Out[14]:
```
approximate_matrix_multiplication (generic function with 1 method)
```

Row-norm sampling

In [15]:
```julia
AT_GA = transpose(A_GA)
dist_GA = DiscreteNonParametric([i for i in 1:n], p_GA)
m = 10000

ATA_GA, frob_GA, spec_GA = approximate_matrix_multiplication(AT_GA, A_GA, dist_GA, m);
```

In [16]:
```
plot( 1:m, frob_GA, label="Frobenius Norm (||·||F) Error",
        box=:on, thickness_scaling=1.2, yscale=:log10)
plot!(1:m, spec_GA, label="Spectral Norm (||·||₂) Error")
title!("AMM | Row Norm Sampling | GA")
xlabel!("Number of Samples [-]")
ylabel!("Normalized Norm Error [-]")
#png("amm_row_norm_sampling_ga.png")
```

Out[16]:

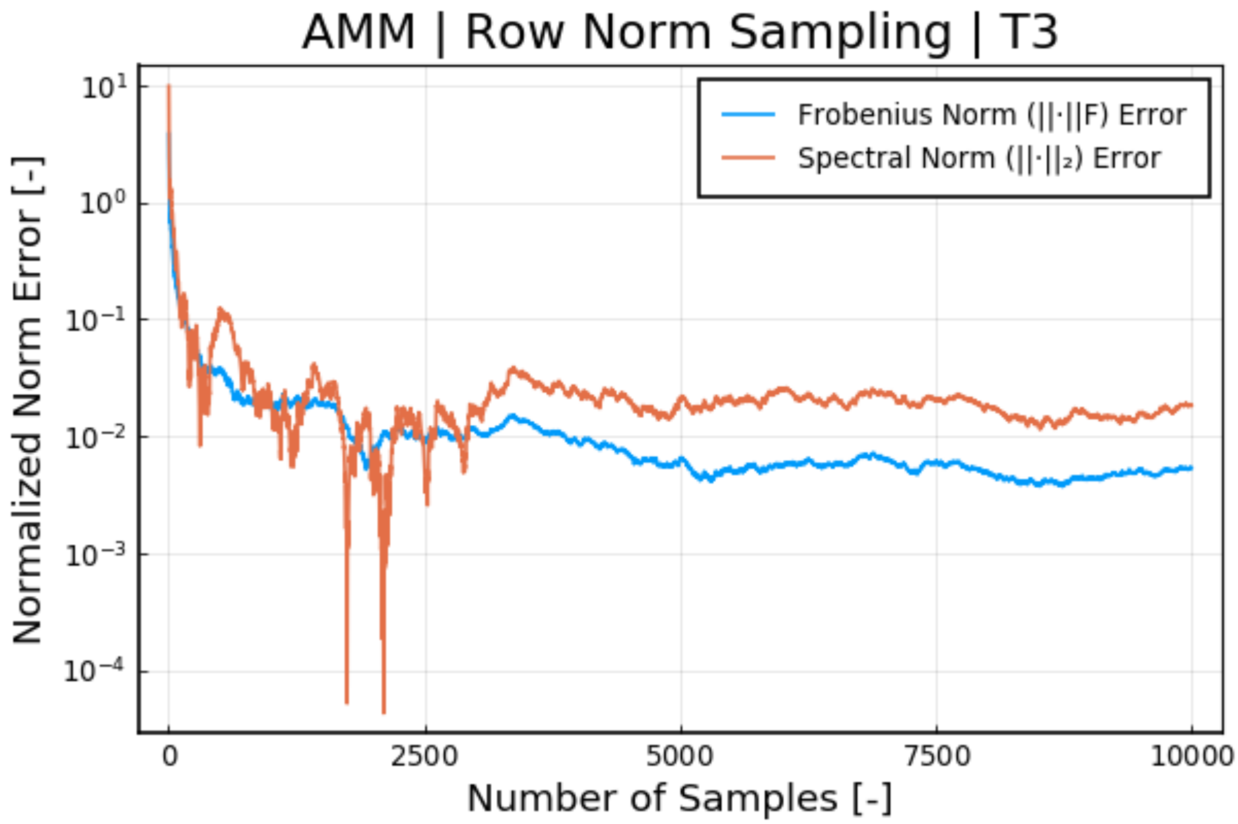


In [17]:
```
AT_T3 = transpose(A_T3)
dist_T3 = DiscreteNonParametric([i for i in 1:n], p_T3)
m = 10000

ATA_T3, frob_T3, spec_T3 = approximate_matrix_multiplication(AT_T3, A_
T3, dist_T3, m);
```

In [18]:

```
plot( 1:m, frob_T3, label="Frobenius Norm (||·||F) Error",
        box=:on, thickness_scaling=1.2, yscale=:log10)
plot!(1:m, spec_T3, label="Spectral Norm (||·||₂) Error")
title!("AMM | Row Norm Sampling | T3")
xlabel!("Number of Samples [-]")
ylabel!("Normalized Norm Error [-]")
#png("amm_row_norm_sampling_t3.png")
```

Out[18]:

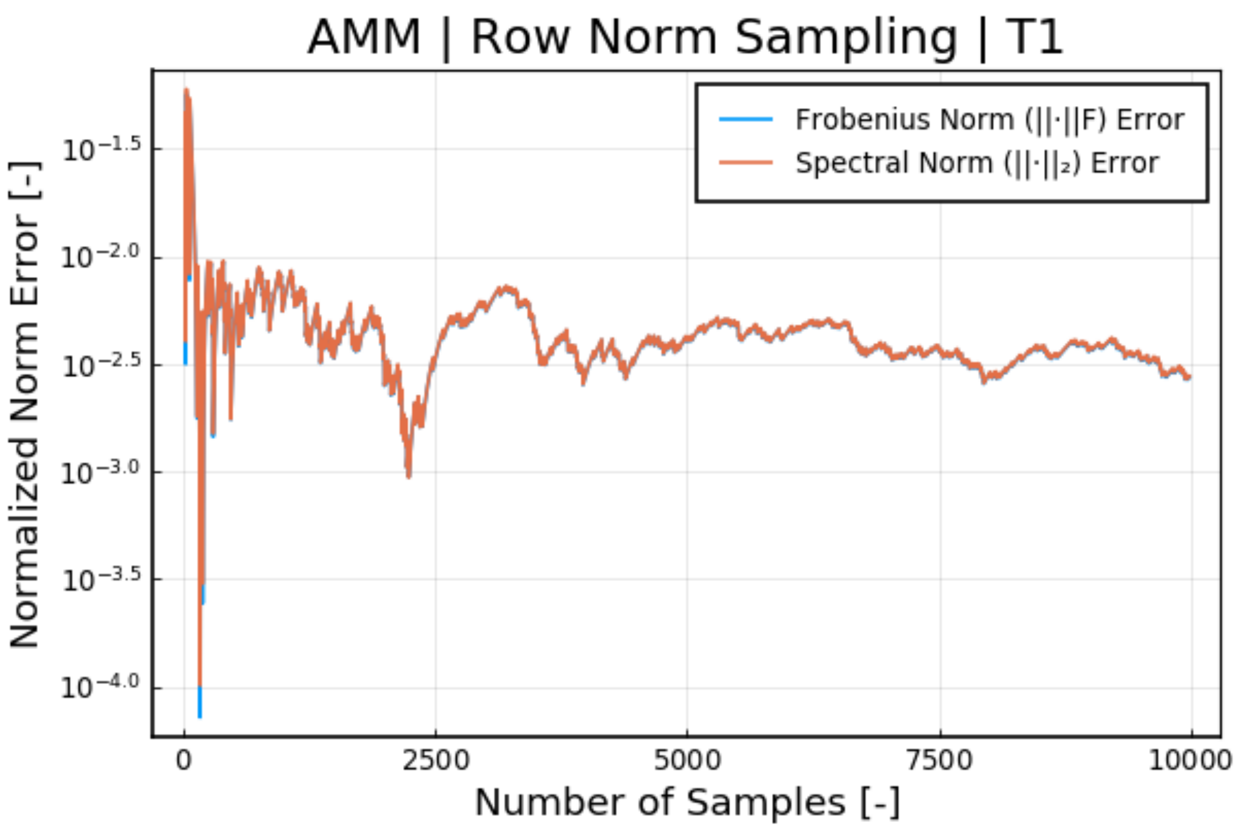


In [19]:

```
AT_T1 = transpose(A_T1)
dist_T1 = DiscreteNonParametric([i for i in 1:n], p_T1)
m = 10000

ATA_T1, frob_T1, spec_T1 = approximate_matrix_multiplication(AT_T1, A_
T1, dist_T1, m);
```

In [20]:
```
plot( 1:m, frob_T1, label="Frobenius Norm (||·||F) Error",
        box=:on, thickness_scaling=1.2, yscale=:log10)
plot!(1:m, spec_T1, label="Spectral Norm (||·||₂) Error")
title!("AMM | Row Norm Sampling | T1")
xlabel!("Number of Samples [-]")
ylabel!("Normalized Norm Error [-]")
#png("amm_row_norm_sampling_t1.png")
```

Out[20]:



Uniform sampling

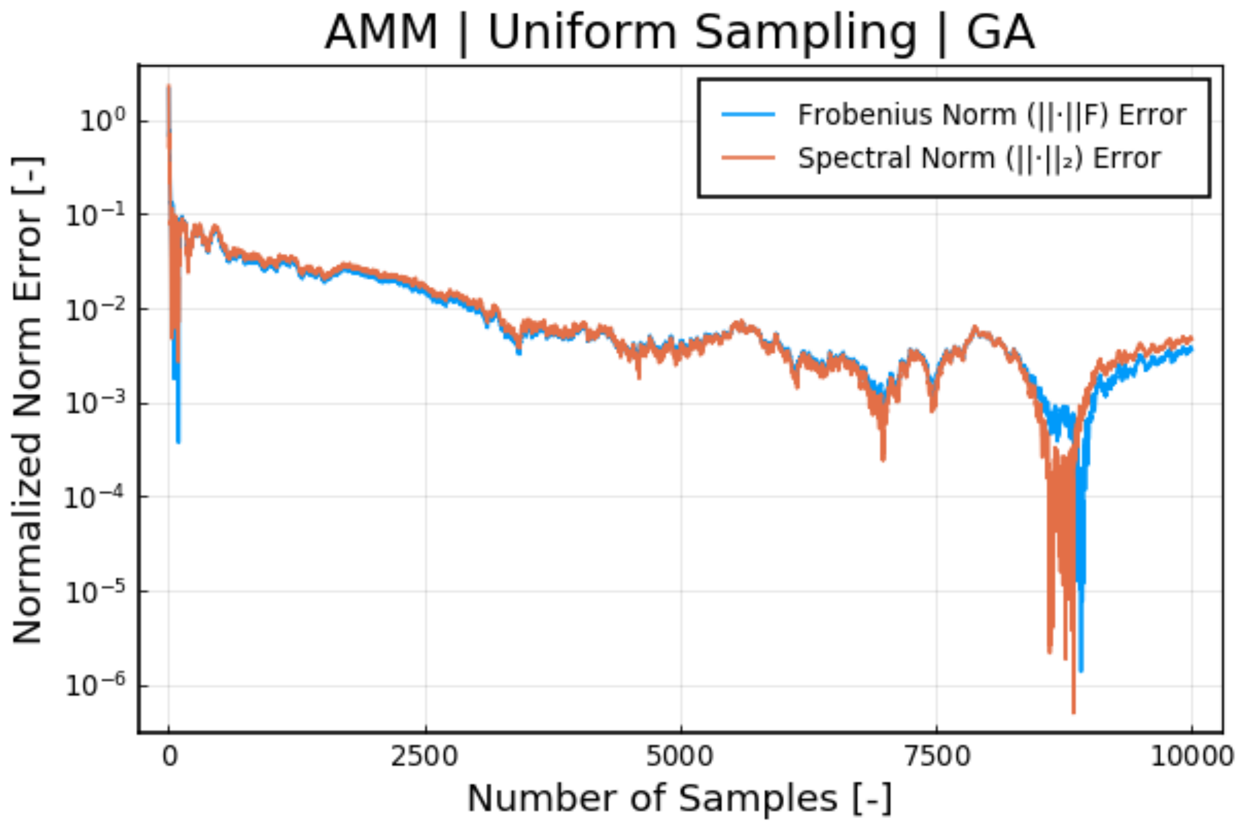
In [21]:
```
uniform_dist = DiscreteNonParametric([i for i in 1:n], [1/n for i in 1
:n])
```

Out[21]:
```
DiscreteNonParametric{Int64,Float64,Array{Int64,1},Array{Float64,1}}
(
support: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10  …  491, 492, 493, 494, 495,
496, 497, 498, 499, 500]
p: [0.002, 0.002, 0.002, 0.002, 0.002, 0.002, 0.002, 0.002, 0.002, 0
.002  …  0.002, 0.002, 0.002, 0.002, 0.002, 0.002, 0.002, 0.002, 0.0
02, 0.002]
)
```

In [22]:
```
ATA_GA, frob_GA, spec_GA = approximate_matrix_multiplication(AT_GA, A_
GA, uniform_dist, m);
```

In [23]:
```
plot( 1:m, frob_GA, label="Frobenius Norm (||·||F) Error",
        box=:on, thickness_scaling=1.2, yscale=:log10)
plot!(1:m, spec_GA, label="Spectral Norm (||·||₂) Error")
title!("AMM | Uniform Sampling | GA")
xlabel!("Number of Samples [-]")
ylabel!("Normalized Norm Error [-]")
#png("amm_uniform_sampling_ga.png")
```

Out[23]:

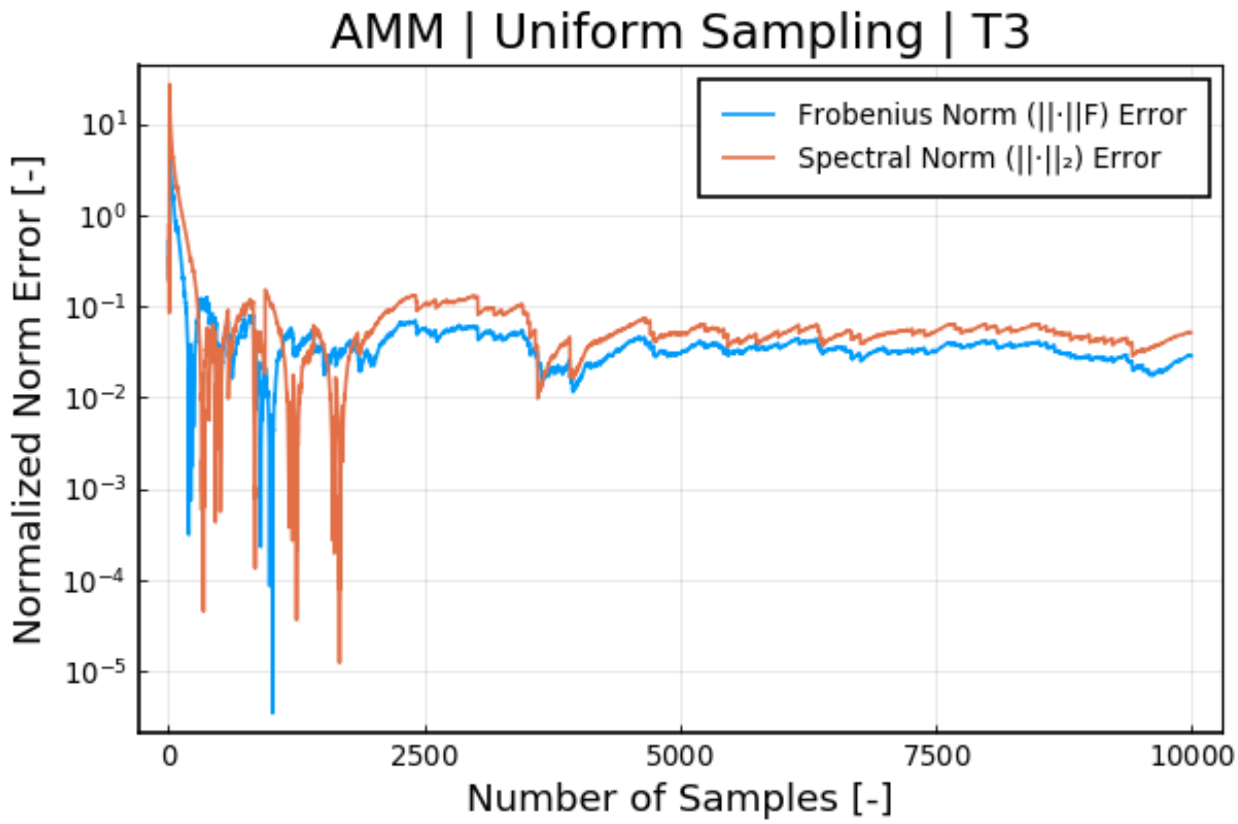


In [24]:
```
ATA_T3, frob_T3, spec_T3 = approximate_matrix_multiplication(AT_T3, A_
T3, uniform_dist, m);
```

In [25]:

```
plot( 1:m, frob_T3, label="Frobenius Norm (||·||F) Error",
        box=:on, thickness_scaling=1.2, yscale=:log10)
plot!(1:m, spec_T3, label="Spectral Norm (||·||₂) Error")
title!("AMM | Uniform Sampling | T3")
xlabel!("Number of Samples [-]")
ylabel!("Normalized Norm Error [-]")
#png("amm_uniform_sampling_t3.png")
```

Out[25]:

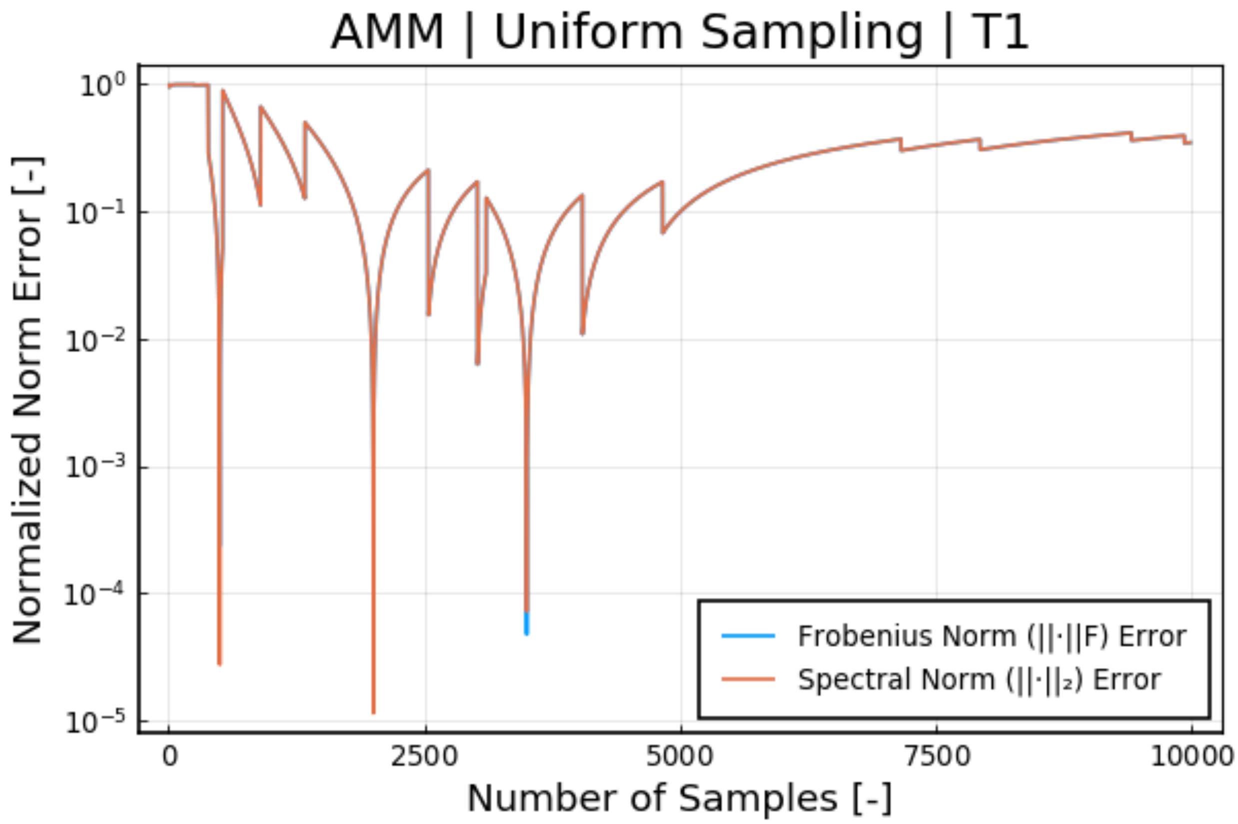


In [26]:

```
ATA_T1, frob_T1, spec_T1 = approximate_matrix_multiplication(AT_T1, A_
T1, uniform_dist, m);
```

In [27]:
```
plot( 1:m, frob_T1, label="Frobenius Norm (||·||F) Error",
        box=:on, thickness_scaling=1.2, yscale=:log10)
plot!(1:m, spec_T1, label="Spectral Norm (||·||₂) Error")
title!("AMM | Uniform Sampling | T1")
xlabel!("Number of Samples [-]")
ylabel!("Normalized Norm Error [-]")
#png("amm_uniform_sampling_t1.png")
```

Out[27]:



AMM on MNIST

In [28]:
```
using MAT
```

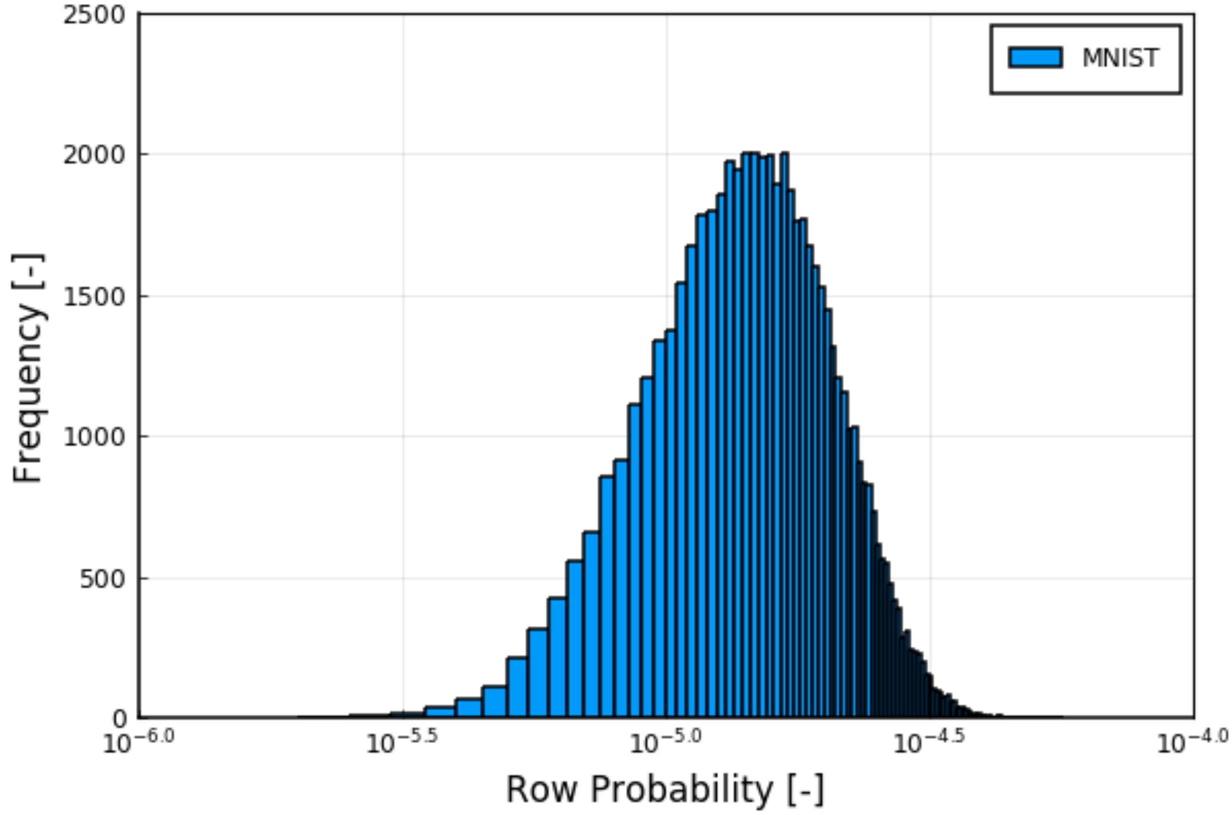
```
WARNING: could not import HDF5.HDF5Group into _hdf5_implementation
WARNING: could not import HDF5.HDF5Dataset into _hdf5_implementation
┌ Warning: Error requiring `HDF5` from `Plots`
│   exception = (LoadError("/Users/rossalexander/.julia/packages/Plo
ts/uCh2y/src/backends/hdf5.jl", 162, UndefVarError(:HDF5Group)), Uni
on{Ptr{Nothing}, Base.InterpreterIP}[Ptr{Nothing} @0x000000010943b7f
f, Ptr{Nothing} @0x00000001094d71d3, Ptr{Nothing} @0x00000001094d905
b, Ptr{Nothing} @0x00000001094d6c5f, Ptr{Nothing} @0x00000001094d6ed
c, Base.InterpreterIP in top-level CodeInfo for Plots._hdf5_implemen
tation at statement 4, Ptr{Nothing} @0x00000001094f0dde, Ptr{Nothing
} @0x00000001094efed0, Ptr{Nothing} @0x00000001094f05e1, Ptr{Nothing
} @0x00000001094f0ce6, Ptr{Nothing} @0x00000001094c9e77, Ptr{Nothing
} @0x00000001094f1ced, Ptr{Nothing} @0x000000010e360c73, Ptr{Nothing
} @0x0000000134a72a7c, Ptr{Nothing} @0x00000001094d907f, Ptr{Nothing
} @0x00000001094d6c5f, Ptr{Nothing} @0x00000001094d6edc, Base.Interp
reterIP in top-level CodeInfo for Plots at statement 10, Ptr{Nothing
} @0x00000001094f0dde, Ptr{Nothing} @0x00000001094f1b07, Ptr{Nothing
} @0x00000001471a3c1f, Ptr{Nothing} @0x00000001471a3c3c, Ptr{Nothing
} @0x0000000134a5cc50, Ptr{Nothing} @0x00000001471a3b7d, Ptr{Nothing
} @0x00000001471a3b9c, Ptr{Nothing} @0x000000011aeea4ab, Ptr{Nothing
} @0x00000001471a3ad3, Ptr{Nothing} @0x00000001471a3afc, Ptr{Nothing
} @0x00000001094ceac8, Ptr{Nothing} @0x00000001094ced85, Ptr{Nothing
} @0x000000011ae950d1, Ptr{Nothing} @0x00000001094ceac8, Ptr{Nothing
} @0x00000001094ced85, Ptr{Nothing} @0x0000000134a6822a, Ptr{Nothing
} @0x0000000134a68ca2, Ptr{Nothing} @0x0000000134a6e57f, Ptr{Nothing
} @0x0000000134a5d3f2, Ptr{Nothing} @0x00000001470bb3c5, Ptr{Nothing
} @0x00000001094f1899, Ptr{Nothing} @0x00000001094f074b, Ptr{Nothing
} @0x00000001094d6bc7, Ptr{Nothing} @0x00000001094d6edc, Base.Interp
reterIP in top-level CodeInfo for Main at statement 0, Ptr{Nothing}
@0x00000001094f0dde, Ptr{Nothing} @0x00000001094f1b07, Ptr{Nothing}
@0x0000000134a9af01, Ptr{Nothing} @0x00000001470ede46, Ptr{Nothing}
@0x00000001094ceac8, Ptr{Nothing} @0x00000001094ced85, Ptr{Nothing}
@0x000000011ae46bef, Ptr{Nothing} @0x000000011ae47164, Ptr{Nothing}
@0x000000011ae4717c, Ptr{Nothing} @0x00000001094dd12a])
└ @ Requires /Users/rossalexander/.julia/packages/Requires/035xH/src
/require.jl:44
```

In [29]:

```
file = matopen("mnist_matrix.mat")
A_MNIST = read(file, "A")
close(file)
```

In [30]:
```
p_MNIST = row_score_probabilities(A_MNIST)

histogram(p_MNIST, label="MNIST",
            xscale=:log10, xlims=(1E-6, 1E-4), ylims=(0, 2500), thickn
ess_scaling=1.1, box=:on,
            xlabel="Row Probability [-]", ylabel="Frequency [-]")
#png("row_score_dist_mnist.png")
```

Out[30]:



In [31]:
```
AT_MNIST = transpose(A_MNIST)
dist_MNIST = DiscreteNonParametric([i for i in 1:size(A_MNIST)[1]], p_
MNIST)
m = 1000

ATA_MNIST, frob_MNIST, spec_MNIST = approximate_matrix_multiplication(
AT_MNIST, A_MNIST, dist_MNIST, m);
```

In [32]:
```
plot( 1:m, frob_MNIST, label="Frobenius Norm (||·||F) Error",
        box=:on, thickness_scaling=1.2, yscale=:log10)
plot!(1:m, spec_MNIST, label="Spectral Norm (||·||₂) Error")
title!("AMM | Row Norm Sampling | MNIST")
xlabel!("Number of Samples [-]")
ylabel!("Normalized Norm Error [-]")
#png("amm_row_norm_sampling_mnist.png")
```

Out[32]: