Assignment 1 Solutions

STAT 4110 - Winter 2026

University of Prince Edward Island
School of Mathematical and Computational Sciences

November 23, 2025

Question 1: Multivariate Normal Transformation (2/15 points)

Since Z has independent standard normal components, we have

$$Z \sim N(0, I_d),$$

where I_d is the $d \times d$ identity matrix. The expectation of Y is

$$\mathbb{E}[Y] = \mathbb{E}[AZ + \mu] = A\mathbb{E}[Z] + \mu = 0 + \mu = \mu.$$

Next, compute the covariance matrix:

$$\mathrm{Cov}(Y) = \mathbb{E}\left[(Y-\mu)(Y-\mu)^T\right] = \mathbb{E}\left[(AZ)(AZ)^T\right] = \mathbb{E}\left[AZZ^TA^T\right] = A\,\mathbb{E}[ZZ^T]\,A^T.$$

Since $Z \sim N(0, I_d)$, we have

$$\mathbb{E}[ZZ^T] = I_d,$$

so

$$\mathrm{Cov}(Y) = AI_dA^T = AA^T = \Sigma.$$

Thus,

$$Y \sim N(\mu, \Sigma).$$

□

Solution to Question 2: Positive Definite Matrices (3/15 points)

Let $A \in \mathbb{R}^{n\times n}$ be symmetric. We prove parts (a)–(c) below.

(a) Invertibility and determinant condition. Assume A is positive definite. We use the quadratic-form characterization of positive definiteness (this is justified in part (b) below): for every nonzero $x \in \mathbb{R}^n$,

$$x^TAx > 0.$$

First show $\ker(A) = \{0\}$. If $v \in \ker(A)$, then $Av = 0$ and thus

$$v^TAv = v^T0 = 0$$

By positive definiteness this implies $v = 0$. Hence A has trivial nullspace, so A is injective. Since A is a linear map on a finite-dimensional space, injectivity implies invertibility. Therefore A is nonsingular.

Next show $\det(A) > 0$. Because A is symmetric, the Spectral Theorem applies: there exists an orthogonal matrix Q and a real diagonal matrix $D = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$ with

$$A = QDQ^T$$

where the λ_i are the eigenvalues of A. For any eigenvector v with eigenvalue λ,

$$v^T Av = v^T(\lambda v) = \lambda\|v\|^2$$

By positive definiteness the left-hand side is > 0 for $v \neq 0$, hence $\lambda > 0$. Thus every eigenvalue $\lambda_i > 0$. The determinant equals the product of eigenvalues:

$$\det(A) = \prod_{i=1}^{n} \lambda_i > 0$$

This proves part (a). $\square$

(b) Quadratic-form characterization. We prove the equivalence:

$$A \text{ is positive definite} \quad \Longleftrightarrow \quad x^T Ax > 0 \text{ for all } x \neq 0$$

There are two common but equivalent definitions of "positive definite" in the symmetric-matrix context; here we show the following implication chain which establishes equivalence when A is symmetric.

(i) $\Rightarrow$ (quadratic form). Suppose all eigenvalues of A are strictly positive (this is a common spectral definition of positive definiteness for symmetric matrices). By the Spectral Theorem $A = QDQ^T$ with $D = \operatorname{diag}(\lambda_1, \ldots, \lambda_n)$, $\lambda_i > 0$. For any $x \neq 0$ set $y = Q^T x$. Then

$$x^T Ax = x^T QDQ^T x = y^T Dy = \sum_{i=1}^{n} \lambda_i y_i^2$$

Each term $\lambda_i y_i^2 \geq 0$ and at least one $y_i \neq 0$ because $x \neq 0$; since each $\lambda_i > 0$ the sum is strictly positive. Thus $x^T Ax > 0$.

(ii) $\Leftarrow$ (quadratic form) Assume $x^T Ax > 0$ for every nonzero x. Let v be an eigenvector of A with eigenvalue λ. Then

$$v^T Av = v^T(\lambda v) = \lambda\|v\|^2$$

Since $v \neq 0$ and the quadratic form is positive on nonzero vectors, the left-hand side is > 0, hence $\lambda\|v\|^2 > 0$. Therefore $\lambda > 0$. Because every eigenvalue arises from some eigenvector, all eigenvalues are positive.

Combining the two implications shows the equivalence: for symmetric A, A has all eigenvalues positive if and only if $x^T Ax > 0$ for all $x \neq 0$. This justifies using the quadratic-form condition as the definition of positive definiteness in the rest of the problem. $\square$

(c) Principal submatrix condition.

Let A be SPD. For $r = 1, \ldots, n$ denote by $A^{(r)}$ the leading $r \times r$ principal submatrix of A; that is,

$$A^{(r)} = \big(a_{ij}\big)_{1 \leq i,j \leq r}$$

We claim each $A^{(r)}$ is positive definite, hence $\det(A^{(r)}) > 0$ by part (a).

Proof: Fix r and take any nonzero vector $z \in \mathbb{R}^r$. Define $x \in \mathbb{R}^n$ by

$$x = \begin{bmatrix} z \\ 0 \end{bmatrix}$$

i.e. x has the first r entries equal to z and the remaining entries equal to 0. Because $A^{(r)}$ is the leading block of A, we have

$$x^T Ax = \begin{bmatrix} z^T & 0 \end{bmatrix} \begin{bmatrix} A^{(r)} & * \\ * & * \end{bmatrix} \begin{bmatrix} z \\ 0 \end{bmatrix} = z^T A^{(r)} z$$

Since A is positive definite and $x \neq 0$ (because $z \neq 0$), it follows that

$$z^T A^{(r)} z = x^T Ax > 0$$

Thus $A^{(r)}$ is positive definite. By part (a) every positive definite matrix is invertible and has positive determinant; therefore

$$\det\big(A^{(r)}\big) > 0$$

This proves the desired implication: if A is positive definite then every leading principal submatrix has positive determinant. $\square$

Question 3: Discrete Random Sampling

Assume that the target distribution satisfies $P(X = x_i) = p_i$ with $\sum_i p_i = 1$. We divide the interval $[0, 1]$ into subintervals such that the i-th subinterval has length p_i. If a generated uniform random variable $U \sim \text{Uniform}[0, 1]$ falls in the j-th subinterval, we assign $X = x_j$.

Mathematically, $X = x_j$ if

$$\sum_{i=1}^{j-1} p_i \le U < \sum_{i=1}^{j} p_i,$$

which is equivalent to saying that $X = x_j$ if j is the smallest integer satisfying

$$U < \sum_{i=1}^{j} p_i.$$

For the given discrete distribution

$$p_i = p(X = i) = \frac{1}{\sqrt[3]{i}} - \frac{1}{\sqrt[3]{i+1}},$$

we have

$$\sum_{i=1}^{j} p_i = \sum_{i=1}^{j} \left(\frac{1}{\sqrt[3]{i}} - \frac{1}{\sqrt[3]{i+1}} \right) = 1 - \frac{1}{\sqrt[3]{j+1}}.$$

The condition $U < 1 - \frac{1}{\sqrt[3]{j+1}}$ can be rewritten as

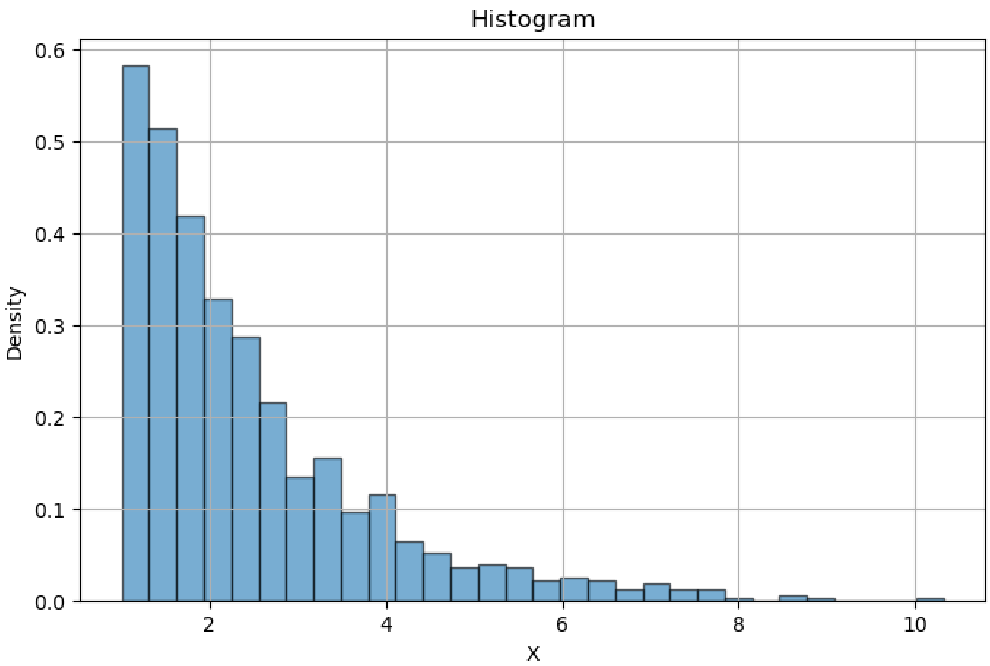
$$\frac{1}{\sqrt[3]{j+1}} < 1 - U \quad \Rightarrow \quad \frac{1}{j+1} < (1-U)^3 \quad \Rightarrow \quad j > \frac{1}{(1-U)^3} - 1.$$

Thus, we define $X = x_j = j$ as the smallest integer satisfying this condition:

$$X = \left\lceil \frac{1}{(1-U)^3} - 1 \right\rceil,$$

where $\lceil \cdot \rceil$ denotes the ceiling function.

```
# parameters
n = 1000
U = np.random.uniform(0, 1, size=n)
X_cont = 1 - np.log2(1 - U)

# histogram
import matplotlib.pyplot as plt
plt.hist(X_cont, bins=30, density=True, alpha=0.7, color='skyblue')
plt.xlabel('X')
plt.ylabel('Density')
plt.title('Histogram')
plt.show()
```



Question 4: Inverse Transform Sampling

We are given the density function

$$f(x) = C \cdot 2^{-x}, \qquad x \geq 1$$

and $f(x) = 0$ otherwise. We determine C, derive the sampling formula, and generate a sample of size $n = 1000$.

Finding the Constant C

We require

$$\int_1^\infty f(x)\,dx = 1.$$

Thus,

$$\int_1^\infty C2^{-x}\,dx = C\int_1^\infty 2^{-x}\,dx = 1.$$

Rewrite using exponentials:

$$\int_1^\infty 2^{-x}dx = \int_1^\infty e^{-x\ln 2}dx = \left[-\frac{e^{-x\ln 2}}{\ln 2}\right]_1^\infty = \frac{e^{-\ln 2}}{\ln 2} = \frac{2^{-1}}{\ln 2} = \frac{1}{2\ln 2}.$$

Thus,

$$C \cdot \frac{1}{2\ln 2} = 1 \quad \Rightarrow \quad \boxed{C = 2\ln 2}.$$

So the density becomes

$$f(x) = 2\ln 2 \cdot 2^{-x}, \qquad x \geq 1.$$

Deriving the Inverse CDF

Compute the CDF:

$$F(x) = \int_1^x 2\ln 2 \cdot 2^{-t}\, dt.$$

Using the earlier integral,

$$F(x) = 2\ln 2 \cdot \frac{2^{-1} - 2^{-x}}{\ln 2} = 2(2^{-1} - 2^{-x}) = 1 - 2^{1-x}.$$

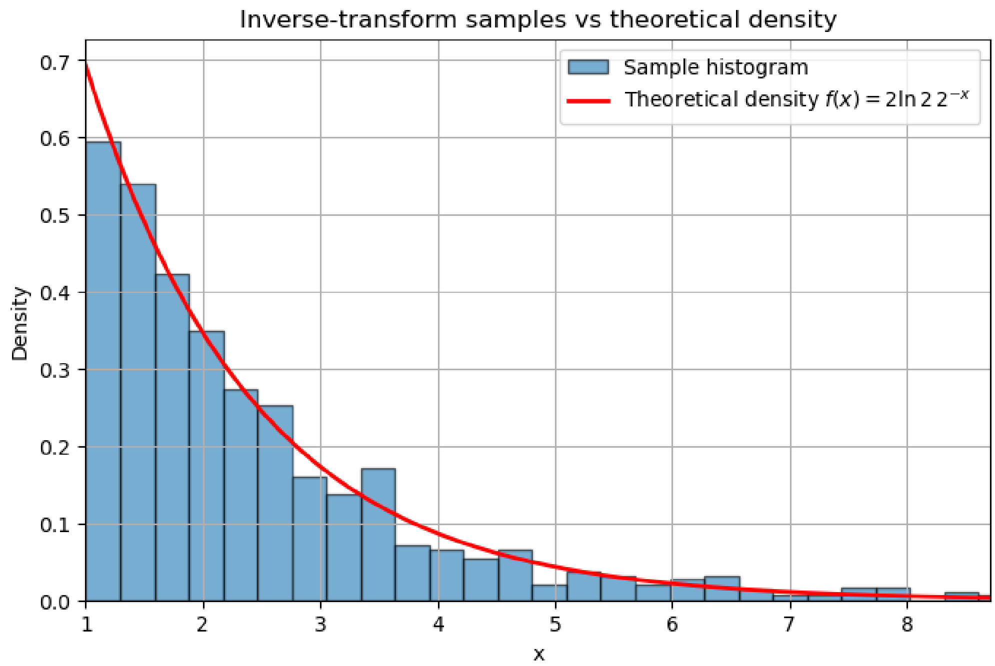
Let $U \sim \text{Uniform}(0,1)$ and set $F(X) = U$:

$$U = 1 - 2^{1-X} \Rightarrow 2^{1-X} = 1 - U \Rightarrow 1 - X = \log_2(1-U) \Rightarrow \boxed{X = 1 - \log_2(1-U)}.$$

In natural log form (numerically stable):

$$\boxed{X = 1 - \frac{\ln(1-U)}{\ln 2}}.$$

```python
# parameters
n = 1000
rng = np.random.default_rng(2025)
ln2 = np.log(2.0)
C = 2.0 * ln2  # C = 2 ln 2

# inverse transform sampling
U = rng.random(n)
X = 1.0 - np.log(1.0 - U) / ln2

# print results
print("Sample mean:", np.mean(X))
print("Sample std:", np.std(X, ddof=1))

# plot
xs = np.linspace(1.0, np.percentile(X, 99.5), 500)
f_x = C * 2.0 ** (-xs)
plt.hist(X, bins=30, density=True, alpha=0.6, edgecolor='black', label='Histogram')
plt.plot(xs, f_x, 'r-', linewidth=2, label='Theoretical Density')
plt.xlabel("x")
plt.ylabel("Density")
plt.title("Inverse Transform Sampling from $f(x) = 2\\ln2 \\cdot 2^{-x}$")
plt.legend()
plt.grid(True)
plt.show()
```



Question 5: Simulation of Bivariate Normal Distribution (4/15 points)

We are given

$$\mu = \begin{bmatrix} 4 \\ 2 \end{bmatrix}, \qquad \Sigma = \begin{bmatrix} 1 & 2 \\ 2 & 9 \end{bmatrix}.$$

We seek a lower-triangular matrix

$$L = \begin{bmatrix} \ell_{11} & 0 \\ \ell_{21} & \ell_{22} \end{bmatrix}$$

with $\ell_{11} > 0,\ \ell_{22} > 0$ such that $LL^T = \Sigma$. Compute entries:

$$LL^T = \begin{bmatrix} \ell_{11}^2 & \ell_{11}\ell_{21} \\ \ell_{11}\ell_{21} & \ell_{21}^2 + \ell_{22}^2 \end{bmatrix} = \begin{bmatrix} 1 & 2 \\ 2 & 9 \end{bmatrix}.$$

Equate entries:

$$\begin{aligned} \ell_{11}^2 = 1 \quad &\Rightarrow \quad \ell_{11} = 1 \text{ (choose positive root)}, \\ \ell_{11}\ell_{21} = 2 \quad &\Rightarrow \quad \ell_{21} = 2, \\ \ell_{21}^2 + \ell_{22}^2 = 9 \quad &\Rightarrow \quad 4 + \ell_{22}^2 = 9 \ \Rightarrow\ \ell_{22}^2 = 5 \ \Rightarrow\ \ell_{22} = \sqrt{5}. \end{aligned}$$

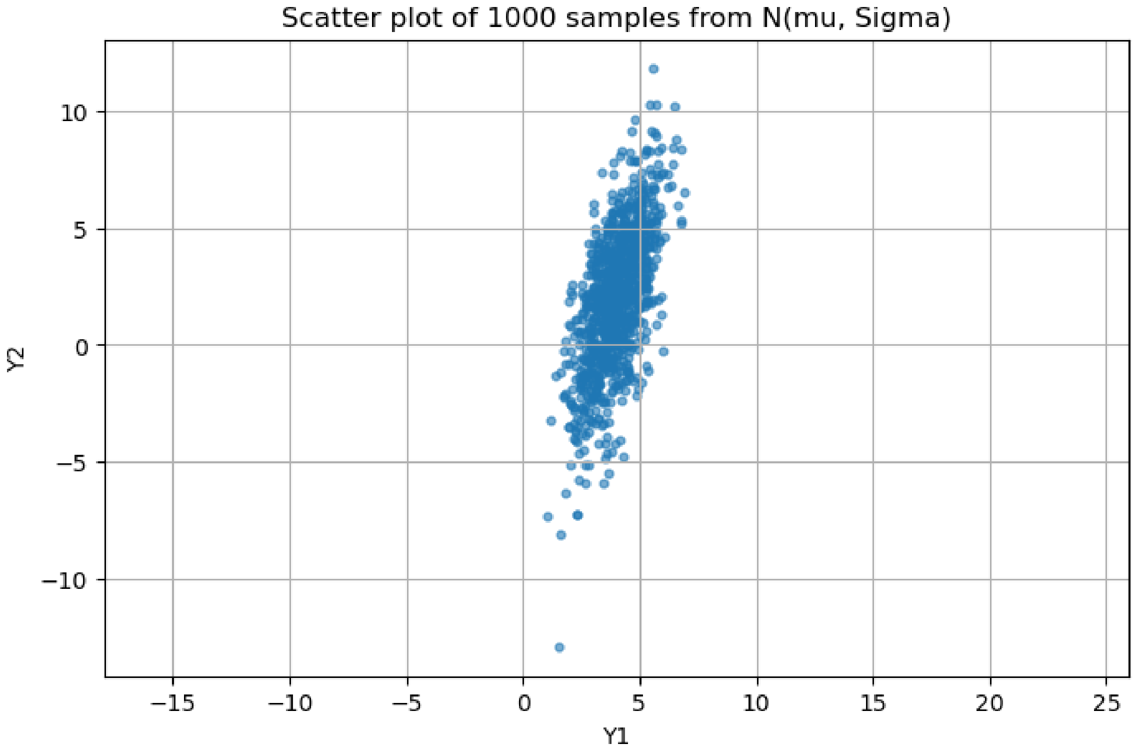
Thus the Cholesky factor is

$$\boxed{L = \begin{bmatrix} 1 & 0 \\ 2 & \sqrt{5} \end{bmatrix}}$$

and one checks $LL^T = \begin{bmatrix} 1 & 2 \\ 2 & 9 \end{bmatrix} = \Sigma$.

Code for Cholesky Decomposition

```python
def cholesky_decomposition(A):
    # initialize an n x n zero matrix to store the lower triangular factor L
    n = len(A)
    L = [[0.0 for _ in range(n)] for _ in range(n)]

    # iterate over each row of the matrix
    for i in range(n):

        # compute the off-diagonal entries of L (columns before the diagonal)
        for j in range(i):
            # compute the summation term
            sum_val = sum(L[i][k] * L[j][k] for k in range(j))

            # compute L[i,j] according to the Cholesky formula
            L[i][j] = (A[i][j] - sum_val) / L[j][j]

        # compute the diagonal entry L[i,i]
        sum_val = sum(L[i][k] ** 2 for k in range(i))

        # take the square root to obtain the diagonal element
        L[i][i] = math.sqrt(A[i][i] - sum_val)

    # convert the list of lists to a NumPy array for easier numerical operations
    return np.array(L)

# parameters
A = [[1, 2], [2, 9]]
mu = np.array([4, 2])
n = 1000

# compute Cholesky factor
L = cholesky_decomposition(A)

# generate standard normals
Z = np.random.standard_normal(size=(n, 2))

# transform to get Y = L Z + mu
Y = Z @ L.T + mu

# sample statistics
sample_mean = np.mean(Y, axis=0)
sample_cov = np.cov(Y, rowvar=False, ddof=1)
print("Sample␣mean:", sample_mean)
print("Sample␣covariance:\n", sample_cov)

# scatter plot
plt.figure(figsize=(8,5))
plt.scatter(Y[:,0], Y[:,1], s=12, alpha=0.6)
plt.xlabel('Y1')
plt.ylabel('Y2')
plt.title('Scatter␣plot␣of␣1000␣samples␣from␣N(mu,␣Sigma)')
plt.grid(True)
plt.axis('equal')
plt.show()
```


Scatter plot of 1000 samples from N(mu, Sigma)
Y2
10
5
0
−5
−10
Y1
−15
−10
−5
0
5
10
15
20
25

Bonus Question (2 points): Determinant Computational Work

We count floating-point operations (flops) where each multiplication or addition is counted as one flop. Use the cofactor expansion along the first row:

$$\det(A) = \sum_{j=1}^{n} (-1)^{1+j} a_{1j} \det(A_{1j})$$

where A_{1j} is the $(n-1) \times (n-1)$ submatrix formed by deleting row 1 and column j.

Let $T(n)$ denote the total number of flops required to compute the determinant of an $n \times n$ matrix by this recursive algorithm (using the cofactor expansion along the first row). We take the base

$$T(1) = 0$$

since a 1×1 determinant requires no arithmetic.

Recurrence for $T(n)$. To compute $\det(A)$ for size $n > 1$ the algorithm:

- recursively computes $\det(A_{1j})$ for each $j = 1, \ldots, n$. Each sub-determinant costs $T(n-1)$ flops, and there are n such subproblems, contributing $n\,T(n-1)$ flops in total;
- multiplies each $\det(A_{1j})$ by the scalar a_{1j}, which costs n multiplications;
- sums the n scalar terms to form the final sum, costing $n-1$ additions.

We can and will ignore the negligible cost of multiplying by the sign $(-1)^{1+j}$. Hence, for $n \geq 2$,

$$T(n) = n\,T(n-1) + n \; + \; (n-1) \; = \; n\,T(n-1) + (2n-1) \tag{1}$$

Set

$$T(n) = n!\, s_n.$$

Substitute into (1). For $n \geq 2$,

$$n!\, s_n = n\big((n-1)!\, s_{n-1}\big) + (2n-1) = n!\, s_{n-1} + (2n-1)$$

Divide by $n!$:

$$s_n = s_{n-1} + \frac{2n-1}{n!}$$

With base $T(1) = 0$ we have $s_1 = T(1)/1! = 0$. Iterating this recurrence for $n \geq 2$ gives

$$s_n = \sum_{k=2}^{n} \frac{2k-1}{k!}$$

Therefore

$$T(n) = n! \sum_{k=2}^{n} \frac{2k-1}{k!}$$

Simplify the inner term

$$\frac{2k-1}{k!} = \frac{2k}{k!} - \frac{1}{k!} = \frac{2}{(k-1)!} - \frac{1}{k!}$$

Hence

$$\sum_{k=2}^{n} \frac{2k-1}{k!} = 2\sum_{k=2}^{n} \frac{1}{(k-1)!} - \sum_{k=2}^{n} \frac{1}{k!}$$

Change indices:

$$\sum_{k=2}^{n} \frac{1}{(k-1)!} = \sum_{m=1}^{n-1} \frac{1}{m!} = \Big(\sum_{m=0}^{n-1} \frac{1}{m!}\Big) - 1$$

and

$$\sum_{k=2}^{n} \frac{1}{k!} = \Big(\sum_{k=0}^{n} \frac{1}{k!}\Big) - 2$$

Let $e_n := \sum_{m=0}^{n} \frac{1}{m!}$ denote the n-th partial sum of the exponential series. Then

$$\sum_{k=2}^{n} \frac{2k-1}{k!} = 2(e_{n-1} - 1) - (e_n - 2) = 2e_{n-1} - 2 - e_n + 2 = e_{n-1} - \frac{1}{n!}$$

since $e_n = e_{n-1} + \frac{1}{n!}$.

Thus

$$T(n) = n! \left(e_{n-1} - \frac{1}{n!}\right) = n!\, e_{n-1} - 1 \tag{2}$$

As $n \to \infty$, the partial sums e_{n-1} converge to e. Therefore

$$T(n) = n!\, e_{n-1} - 1 \sim e\, n! \qquad \text{as } n \to \infty$$

So the leading asymptotic behaviour of the flop count for the naive recursive determinant is

$$T(n) \approx e\, n!$$