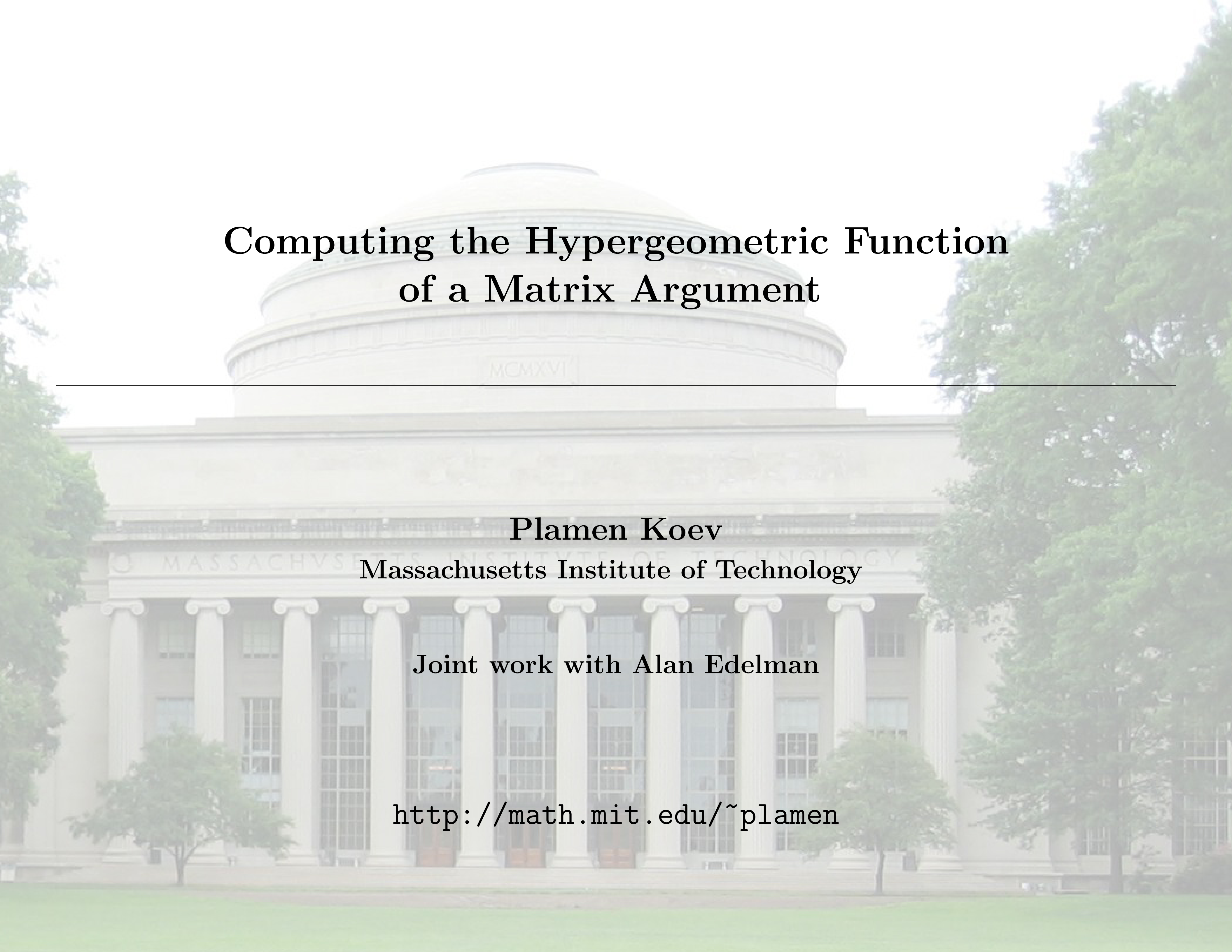


Computing the Hypergeometric Function of a Matrix Argument

Plamen Koev
Massachusetts Institute of Technology

Joint work with Alan Edelman

`http://math.mit.edu/~plamen`

Why ${}_pF_q(\cdot,\cdot,X)$?

- Distributions of

$$\lambda_{\min}, \lambda_{\max}, \det, \text{ etc.}$$

  of

  Wishart, Jacobi, Laguerre

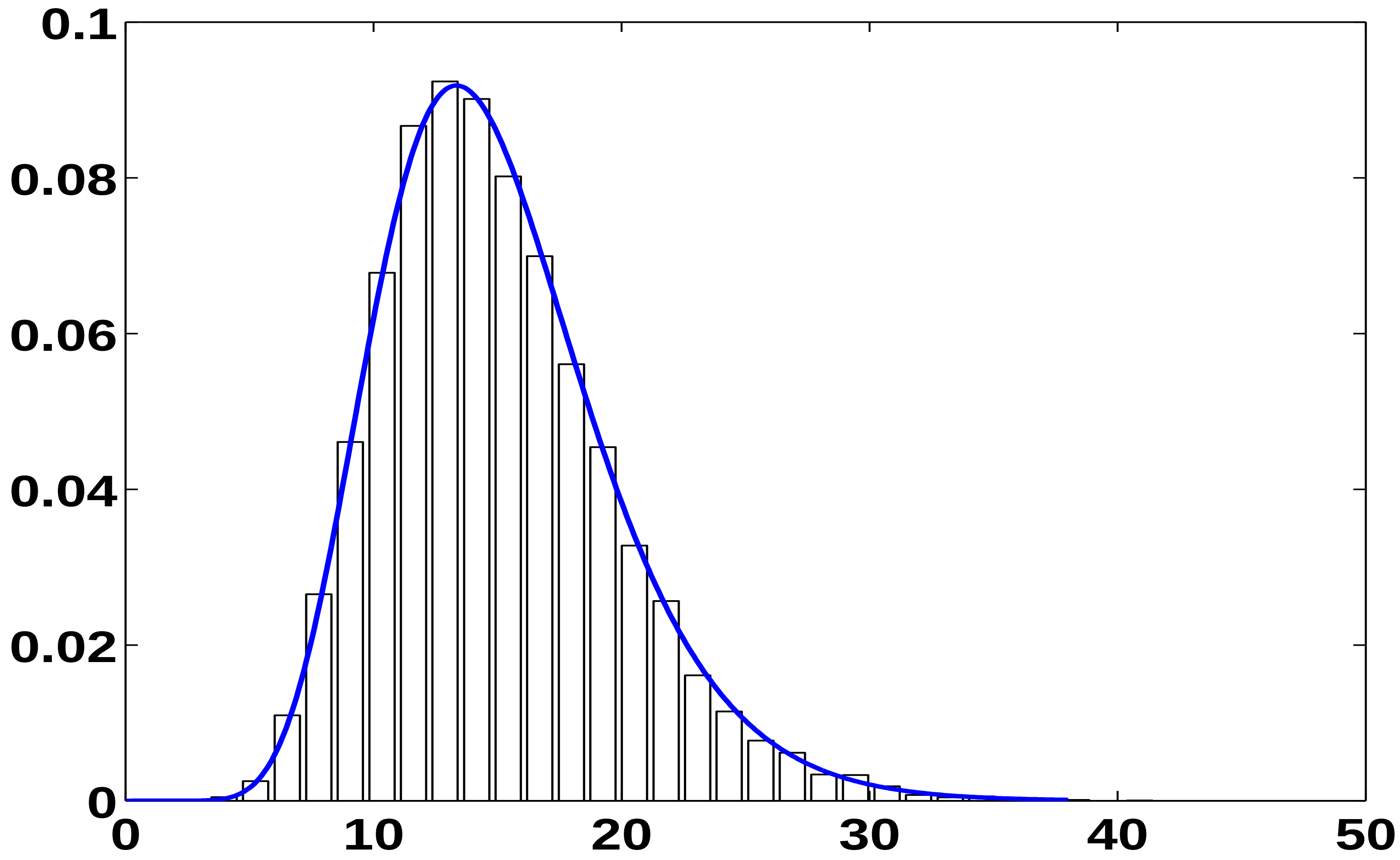
  expressed in terms of ${}_pF_q(\cdot,\cdot,X)$

- The distributions useful in:
  - Hypothesis testing (e.g., $\Sigma = I$, etc.)
  - Parameter estimation: $A \sim W_m(n, \sigma^2 I), \quad \sigma = ?$

- Applications in:
  - Population classification
  - Automatic target classification
  - Wireless communications

- Computing ${}_pF_q(\cdot,\cdot,X)$: 40-year-old open problem
  - Notorious complexity and slow convergence
  - Empirical methods inefficient

Distribution of λ_{max} of 4×4 Wishart with 7 DOF, $\Sigma = I$



- Exact vs Empirical with 20,000 replications

James 1964

If $A \sim W_n(m, \Sigma)$ then

$$P(\lambda_{\max}(A) < x) \sim x^{\frac{m}{2}} \cdot {}_1F_1\left(\tfrac{m}{2}; \tfrac{n+m+1}{2}; -\tfrac{1}{2}x\Sigma^{-1}\right)$$

$$= x^{\frac{m}{2}} \cdot \sum_{k=0}^{\infty} \sum_{\kappa \vdash k} p_\kappa \cdot x^k \cdot C_\kappa(\Sigma^{-1})$$

- Slow convergence $\Rightarrow$ $\infty \sim 50, 100, 150$
- $C_\kappa(X)$ – *Zonal Polynomial* – Really hard: $O(n^m)$ terms in each!
- Our Contribution: $O(n)$
- Impossible until now
  - Previous best algorithm ($n = 5$): 8 days

    (Gutiérrez, Rodriguez, Sáez, 2000)
  - New algorithm: $\frac{1}{100}$ second

Automatic 3D Target Classification

Blazer

X_1

HMMWV

X_2

M1 A1 Abrams

X_3

Leopard

X_4

T62

X_5

Challenger

X_6

- X_i — $n \times 3$ matrices (given data)
- Observation: $X = (X_i + E)Q, \qquad e_{ij} \sim N(0, \sigma^2)$
- $X^T X$ — noncentral Wishart, eigs do not depend on Q
- Question: $i = ?$

$$L(i|X) \sim {}_1F_0(\cdot; X^T X)$$

- Reference: Michael Jeffris (MITRE), Proceedings of SPIE 2005

Computing ${}_pF_q(\cdot;\cdot;X)$

$$P(\lambda_{\max}(A) < x) \sim x^{\frac{m}{2}} \cdot {}_1F_1\left(\tfrac{m}{2}; \tfrac{n+m+1}{2}; -\tfrac{1}{2}x\Sigma^{-1}\right)$$
$$= x^{\frac{m}{2}} \cdot \sum_{k=0}^{\infty} \sum_{\kappa \vdash k} p_\kappa \cdot x^k \cdot C_\kappa(\Sigma^{-1})$$

- Means computing zonal polynomials $C_\kappa(\Sigma^{-1})$
- $C_\kappa(\Sigma^{-1})$ depends only on the eigenvalues $x_1, x_2, \ldots, x_n$ of Σ^{-1}
- Illustrate $\beta = 2$ (complex); general β analogous

Partition κ	C_κ	Number of terms
(1)	$x_1 + \cdots + x_n$	$O(n)$
(2)	$\sum_{i \le j} x_i x_j$	$O(n^2)$
$(1,1,1)$	$\sum_{i<j<k} x_i x_j x_k$	$O(n^3)$
κ	$\sum_T x^T$	$O\left(n^{\lvert\kappa\rvert}\right)$

Computing $C_\kappa(X)$

IDEA: C_κ **are** $\chi(\mathrm{GL}_n(\mathbb{C}))$; $\chi(\mathrm{GL}_{n-1}(\mathbb{C}))$ **induce** $\chi(\mathrm{GL}_n(\mathbb{C}))$

Example:

$$C_{(1,1)}(X) = \sum_{i<j} x_i x_j = x_1x_2 + (x_1+x_2)x_3 + \cdots + (x_1 + \cdots + x_{n-1})x_n$$

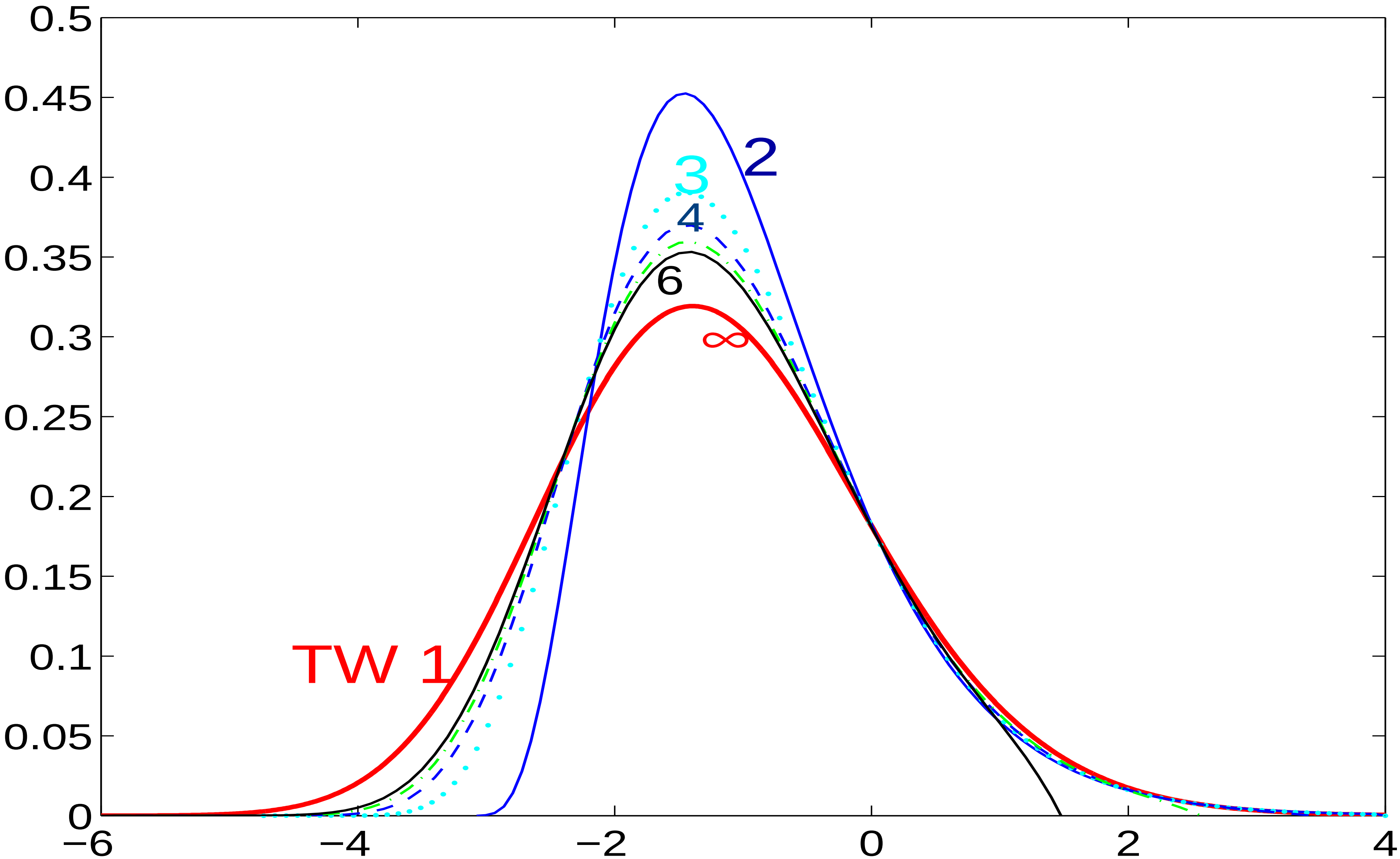
Algorithm:

$$\begin{aligned} s_1 &= x_1 \\ s_2 &= s_1 + x_2 && (= x_1 + x_2) \\ s_3 &= s_2 + x_3 && (= x_1 + x_2 + x_3) \\ &\vdots \\ s_{n-1} &= s_{n-2} + x_{n-1} && (= x_1 + x_2 + \cdots + x_{n-1}) \end{aligned}$$

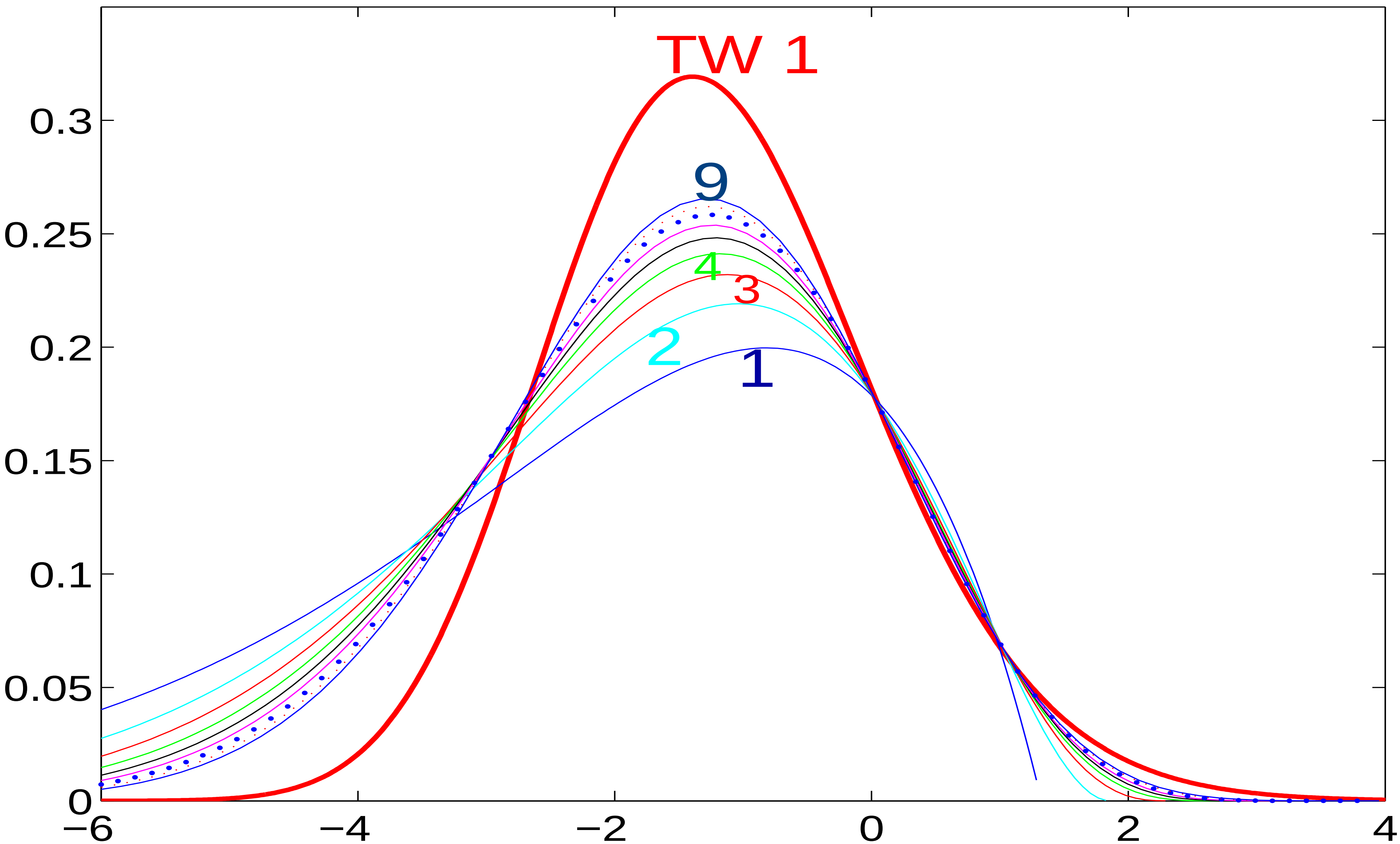
$$C_{(1,1)}(X) = s_1x_2 + s_2x_3 + \cdots + s_{n-1}x_n$$

- **Cost:** $O(n)$ **versus** $O(n^2)$
- **In general:** $O(n)$ **versus** $O(n^{|\kappa|})$

$$A_p \sim W_p(n, I); \quad n/p = 5; \quad (\lambda_{\max}(A_p) - \mu_p)/\sigma_p \to \mathrm{TW}_1$$



Jacobi: $A_p \sim W_p(q,\Sigma),\ B_p \sim W_p(n,\Sigma),\ A_p - \lambda(A_p+B_p)$



$p = 4k+2,\ n/q = 2,\ n/p = 3,\ \lambda_{\max}(A_p B_p^{-1}) \to \lambda_{\max}(A_\infty B_\infty^{-1}) \sim \mathrm{TW}_1$

Future work

- **Incorporate the work of Johnstone, Richards, Chen, Tracy-Widom**
- **New Cooley-Tukey-type algorithm:**
  - $(\mathrm{DFT})_{ij}$—characters of $\mathbb{Z}/n\mathbb{Z}$ $\longleftrightarrow$ C_λ—characters of $\mathrm{GL}_n(\mathbb{C})$
  - Main identity

$$C_\kappa(x_1, x_2, \ldots, x_n) = \sum_{\lambda < \kappa} C_\lambda(x_1, x_2, \ldots, x_{n-1}) \cdot x_n^{|\kappa|-|\lambda|} \cdot f_{\lambda\kappa}^{\alpha}$$

In matrix form:

$$\mathcal{C}_n = \mathcal{C}_{n-1} \cdot Y_n(x_n)$$

where for example

$$Y_2(x) = \left[\begin{array}{cccc|cccc|cccc|cccc} 1 & x & x^2 & x^3 & & & & & & & & & & & & \\ & 1 & x & x^2 & x & x^2 & x^3 & x^4 & & & & & & & & \\ & & 1 & x & & x & x^2 & x^3 & x^2 & x^3 & x^4 & x^5 & & & & \\ & & & 1 & & & x & x^2 & & x & x^2 & x^3 & x^3 & x^4 & x^5 & x^6 \end{array}\right]$$

  - Y_n structured ... MVM takes linear time
  - $\mathrm{Cost(New\ Alg)} \approx \sqrt{\mathrm{Cost(Current\ Alg)}}$ ... just like FFT