Classifying Pseudorandom Number Generators Using Statistical Test Results

Prof. Michael Mascagni

Department of Computer Science
Department of Mathematics
Department of Scientific Computing
Graduate Program in Molecular Biophysics
Florida State University, Tallahassee, FL 32306 **USA**
E-mail: mascagni@fsu.edu
URL: http://www.cs.fsu.edu/∼mascagni

Outline

Motivation

- Pseudorandom number generators (PRNGs) are essential in:
  - Scientific simulation, cryptography, game development, machine learning
- PRNGs use **deterministic algorithms** to produce sequences that emulate randomness.
- Quality is evaluated using standardized test suites — most notably **TestU01**.
- When a PRNG passes these tests, its output is sufficiently random that identifying the underlying algorithm from the output stream alone is **difficult or impossible**.
- **Key question:** Can we identify the PRNG algorithm from its *statistical test results* using machine learning?

Our Approach

Goal

Given a set of TestU01 p-values from an unknown PRNG stream, classify:

1. The **generator algorithm** that produced the stream.
2. The **parameters** used to initialize that generator.

Libraries tested:

- SPRNG (Scalable Parallel RNG)
- TestU01 built-in generators

TestU01 Batteries used:

- Small Crush
- Crush
- Big Crush

Generators Studied (14 Total)

SPRNG Library

- LCG — 48-bit linear congruential
- LCG64 — 64-bit linear congruential
- PMLCG — prime modulus LCG
- CMRG — combined multiple recursive
- LFG — lagged Fibonacci
- MLFG — multiplicative lagged Fibonacci

TestU01 Library

- AES — AES-based generator
- BRENT — Brent's xor generator
- CLCG — combined LCG
- GFSR — generalized feedback shift register
- TGFSR — twisted GFSR
- ISAAC — indirection/shift/accumulate generator
- SHA1 — secure hash algorithm generator
- TAUS — simple Tausworthe generator

Generator Parameter Sets & Test Runs

Generator	Param. Sets	Small Crush	Crush	Big Crush
TAUS	1	64	64	64
PMLCG	1	30	32	100
TGFSR	1	64	64	126
ISAAC	2	64	64	128
CLCG	3	171	171	225
BRENT	2	128	128	251
SHA1	4	128	128	256
CMRG	3	90	96	300
LCG64	3	90	96	300
GFSR	3	192	192	378
AES	6	192	192	384
LCG	7	210	224	700
LFG	11	330	352	1100
MLFG	11	330	352	1100
Total		**2083**	**2155**	**5412**

Table: Number of tests run per generator in each battery.

Data Collection Process

- Data collected on the **Hyperion HPC cluster** at USCB.
- Serial scripts run one battery per generator/parameter/seed combination.
- **Seed generation** — two approaches:
  - Structured sequences varying binary representations:

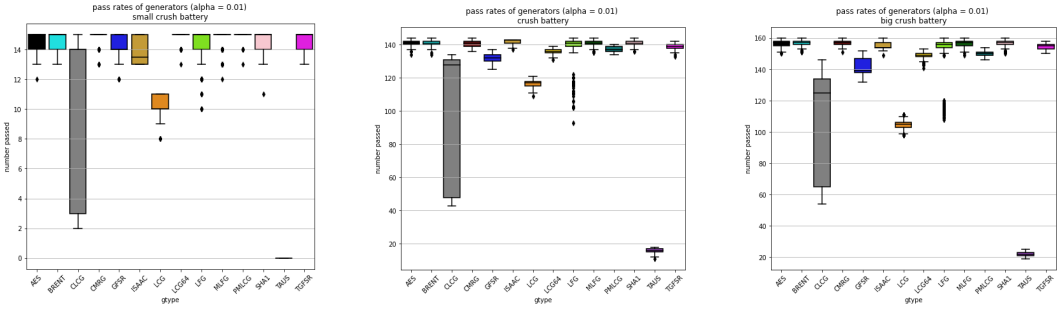
$$x_n = \left((2^n + n^2 + \lfloor n(n+1)/2 \rfloor^2 + \ldots) \oplus x_{n-1}\right) \bmod 2^{32}$$

  - Python `random.randint` for uniform random seeds in $[0, 2^{32} - 1]$.

- **Data cleaning:**
  - Duplicate p-value labels disambiguated.
  - Null/negative values (non-applicable tests) replaced with 0.
  - Features scaled to avoid high-variance dominance.

Pass Rate Distributions

- A test is **passed** when its p-value $> \alpha = 0.01$.
- Most generators show **tight pass-rate distributions** — parameters have little influence.
- Notable exceptions: **CLCG** and **LFG** show wider spreads.



Small Crush | Crush | Big Crush

Figure: Distribution of pass rates across all generators for each test battery.

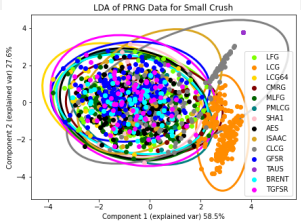
Dimensionality Reduction — Overview

- Two methods applied to each dataset:
  - **PCA** — Principal Component Analysis (unsupervised)
  - **LDA** — Linear Discriminant Analysis (supervised)

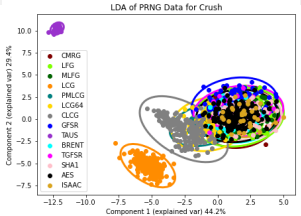
- **Findings from 2-component projection:**
  - Most generators form a large central cluster in 2D.
  - Low-pass-rate generators (**TAUS, LCG, CLCG, GFSR**, LFG subgroup) are **visibly separated**.
  - PCA explained variance is *low* for 2 components — many components needed for a good fit.
  - LDA explains variance better but compresses top-performing generators into a tighter cluster.

- LDA reduced data (13 features) was the **best input for classification models**.

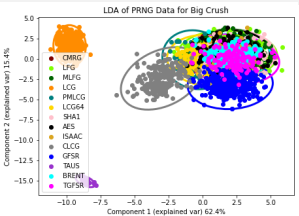
LDA and PCA Projections



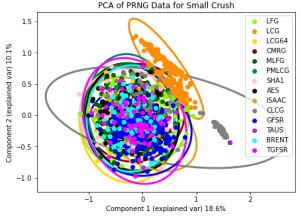
LDA – Small Crush



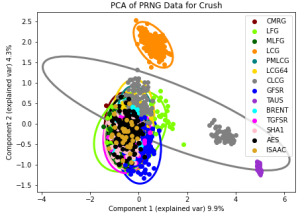
LDA – Crush



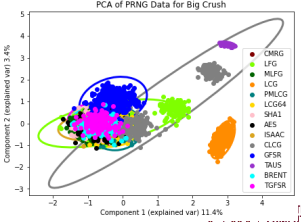
LDA – Big Crush



PCA – Small Crush



PCA – Crush



PCA – Big Crush

Classification Setup

- Multiple supervised classifiers trained on each of the three datasets.
- Default `scikit-learn` hyperparameters; class weights balanced.
- **Evaluated via cross-validation accuracy.**

Classifiers Compared

- Voting Classifier (ensemble)
- One-vs-Rest (OVR)
- Histogram Gradient Boosting
- Bagging Decision Trees
- Random Forest
- LDA + k-Nearest Neighbors

- **Voting Classifier** components: Histogram Gradient Boosting, Bagging Decision Trees, Random Forest, LDA-kNN.

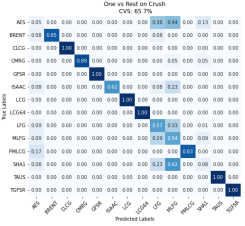
Cross-Validation Accuracy — Generator Classification

Classifier	Small Crush	Crush	Big Crush
Voting	0.443	0.664	**0.707**
One vs Rest (OVR)	0.431	0.657	0.705
Histogram Gradient Boosting	**0.447**	0.657	0.698
Bagging Decision Trees	0.418	0.643	0.698
Random Forest	0.437	**0.645**	0.667
LDA kNN	0.349	0.542	0.569

Table: Cross-validation scores for each classifier across all three batteries.

- Accuracy **improves substantially** with larger test batteries.
- Voting and OVR are top performers at $\approx 70\%$ **on Big Crush**.
- Performance on Small Crush ($\approx 44\%$) reflects limited test information.

Confusion Matrices — Best Classifiers



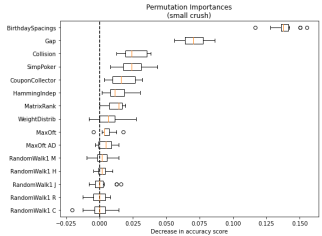
OVR – Crush | Voting – Crush | OVR – Big Crush | Voting – Big Crush

Figure: Confusion matrices for the two best-performing models on Crush and Big Crush.

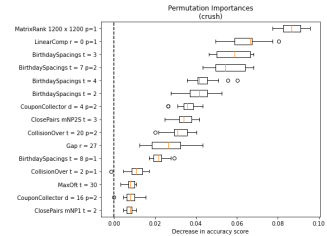
- Common misclassification: TestU01 **ucrypto** generators confused with **MLFG**.
- A subset of **LFG** streams also confused with MLFG — likely due to similar statistical footprints.

Permutation Feature Importance

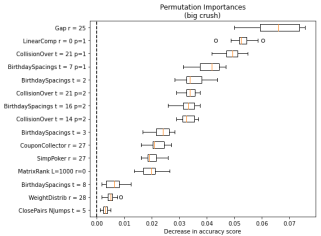
- Used **permutation importance** (scikit-learn) to identify the most influential features.
- Algorithm: shuffle each feature, record accuracy drop — averaged over multiple runs.
- Showing the **top 14 features** for each battery.



Small Crush



Crush



Big Crush

Figure: Top-14 permutation feature importances for the Voting Classifier.

- Consistently high-importance tests: **Gap**, **Birthday Spacings**, **Close Pairs**.

Neural Network Classifier

- A neural network was trained and compared against the ensemble classifiers.
- Dimensionality reduction applied *before* the network:
  - SelectKBest
  - PCA
  - **LDA (best)**
- LDA reduced to **13 features** (from 144–160).

Reduction	**SC**	**C**	**BC**
LDA	**0.367**	**0.661**	**0.629**
KBest (k=55)	N/A	0.580	0.599
PCA	0.348	0.548	0.586
Unreduced	0.353	0.571	0.572

Table: NN accuracy by reduction method.

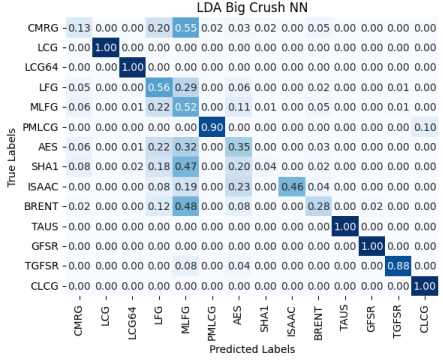


Figure: Confusion matrix for the best NN (Big Crush + LDA).

Classifying Generator Parameters

- Classifiers also trained to predict **which parameter set** was used within each generator.
- (TAUS, PMLCG, TGFSR) exclude as only 1 parameter— 11 generators used.
- Results vary greatly by generator.

Generator	Params	Small Crush	Crush	Big Crush
LCG	7	0.676	0.991	**1.000**
LCG64	3	0.300	**1.000**	**1.000**
CMRG	3	0.333	0.979	0.997
GFSR	3	0.922	0.990	0.995
ISAAC	2	**1.000**	0.985	0.992
CLCG	3	0.924	**1.000**	0.787
LFG	11	0.185	0.284	0.432
BRENT	2	0.079	0.103	0.323
SHA1	4	0.258	0.194	0.278
AES	6	0.167	0.171	0.143
MLFG	11	0.097	0.080	0.093

Table: One-vs-Rest cross-validation accuracy for parameter classification.

Parameter Classification — Key Observations

Strong or perfect predictions:

- **LCG, LCG64, CMRG, GFSR, CLCG** — near-perfect on Crush/Big Crush.
- **ISAAC** — perfect on Small Crush; initialization algorithm differs between parameters.

LFG subgroup:

- 5 of 11 parameter sets form a **separate PCA cluster**.
- These 5 produce **smaller period streams**, distinguishing them statistically.
- Classifier can identify this subgroup reliably.

Poor predictions:

- **AES, MLFG, BRENT, SHA1** — accuracy near or below chance.
- Cryptographic-strength generators (AES, SHA1) produce statistically indistinguishable streams regardless of parameters.
- MLFG parameter sets have similar statistical footprints.

Conclusions

Summary of Findings

- TestU01 p-value vectors carry a **statistical signature** of the generating algorithm.
- Ensemble classifiers (Voting, OVR) achieve $\approx 70\%$ **accuracy** in generator identification on Big Crush data.
- **Non-cryptographic generators** are classified much more reliably than cryptographic ones.
- LDA preprocessing is the most effective dimensionality reduction strategy.
- Parameter classification is **perfect or near-perfect** for several generators (LCG, LCG64, CMRG, GFSR, ISAAC, CLCG) on Crush and Big Crush.
- Key discriminating tests: **Gap**, **Birthday Spacings**, **Close Pairs**.

Future Work

1. **Expand the generator set**
   Include generators outside SPRNG and TestU01 libraries to improve classifier generalization.
2. **Additional statistical test suites**
   Incorporating tests beyond TestU01 may improve discriminative power.
3. **Unsupervised classification**
   Use clustering algorithms to discover natural groupings and compare with supervised results.
4. **Public test-results database**
   Make TestU01 results available online to avoid redundant computation and support the research community.
5. **Scrambled generators**
   Investigate whether non-linear scrambling (e.g., counter-based/Feistel scramblers) changes classification accuracy — and whether scrambled generators remain classified as their underlying algorithm.

Work was done with my student and his students:

W. John Thrasher	thrashw@uscb.edu
Christoph Hagenauer	chhagenauer@icloud.com
Jarrett Kizer	jarrettkizer@gmail.com

Questions?