DYALOG
2020

Time Travel Debugging & Statistical Distributions

Ron Murray

Presentation Outline

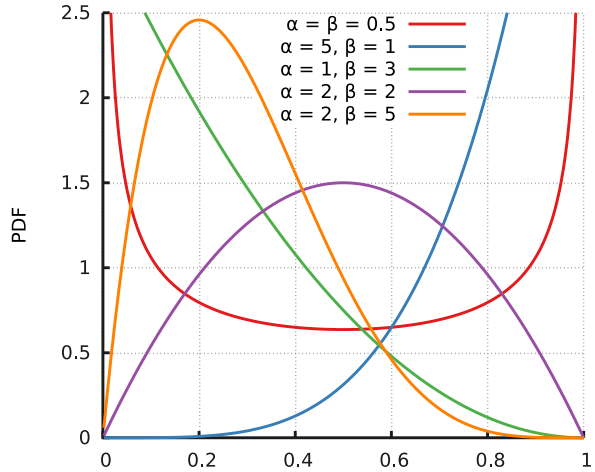
- Non-Linear Random Number Distributions
  - Why they Matter
  - Current Strategies
  - What we are adding for 19.0
- What We're Doing about Testing APL
  - Quantifying Code Coverage
  - Literate Unit Tests
- Some Personal History
  - APL 1969-1986
  - Non-APL 1986-2019

Non-Linear Distributions – Why?

- Ulam's Monte Carlo Strategies
  - Using random sampling to model physical processes
    - In physics (Neutron defusion and chain reactions) some things are probabilistic and hard to calculate.
    - Instead Use Non-Linear Probabilistic Models
    - Sample the Space and Gather Statistics
  - Applied now to many physical, financial, and management problems

Current Strategies

- Beta Distribution is non-linear and non-uniform!



Current Strategies – Mapping from ?0

- Beta Parameters: 5 1
  - (?Nρ0)*÷5
- Beta Parameters: 1 5
  - 1-(?Nρ0)*÷5
- Beta Parameters: 1 1
  - ?Nρ0
- Inverse Transform Sampling
- Using R via rconnect.dws

New Support in Version19.0

- `r←2 5(16808⌶) 'Beta' 10000`
- `parameters (16808⌶) TypeName, Shape`
- We will add more distribution types
  - Dirichlet
  - Gamma
  - Normal

  - Suggestions?

New Support in19.0 -- Performance

- Much less space and time than Inverse Transform Method
- Less overhead than starting and communicating with R
- A native implementation in C/C++
- The very same algorithms used by R and other systems
- A clean framework for supporting additional distributions

QA work for APL

- Quantifying Code Test Coverage
- Writing Focused Unit Tests
  - Literate documentation of correct behavior
  - Verification of error detection and signaling
- Close examination of new code
  - Especially “optimizations”!
  - Lots of time debugging failures we find!

QA Work – Writing Unit Tests as Documentation – Part 1

```
passed←Test_with_matched_shapes_and_sorted_left_arg;a;b;r;⎕TRAP
⍝ Environment:
⎕TRAP←0 'C' '→unexpected_signal'

⍝ Invariant under test:
⍝ xs←⍴a ⋄ xr←⍴⍴a ⋄ ys←⍴b ⋄ yr←⍴⍴b
⍝ ((1↓xr) ≡ 1↓yr) ∧ (yr ≥ xr -1) ∧ (1 < xr)
⍝ then a must be sorted in ascending order
⍝ othersize signal domain error

⍝ Given:
a←100 #.test_data.sorted_array_of_positive_integers 3 4 5
b←?6 4 5⍴100
```

QA Work – Writing Unit Tests as Documentation – Part 2

```
⍝ When I:
r←a⍳b
passed←r VerifyIntervalIndexResult⊂a),⊂b
→passed/0

#.failure_logger.recordFailure'Incorrect result' 'r←a⍳b'r a b
→0

unexpected_signal:#.failure_logger.recordFailure'Unexpected signal!'⎕DM'r←a⍳b'⍬ a b
→passed←0
```

QA Work – Writing Unit Tests for Error Behavior

```
passed←Test_for_scalar_left_argument;a;b;r;⎕TRAP
⍝ Environment
⎕TRAP←4 'C' '→rank_error'
⍝ Invariant under test:
⍝ xr←⍴⍴⍺ ⋄ yr←⍴⍴⍵ ⋄ cr←xr-1
⍝ Signals Rank Error unless xr>0 AND yr>=cr

⍝ Given:
a←1
b←⍳5
⍝ When I:
r←a⍳b
#.failure_logger.recordFailure'Should have signaled Rank Error!' 'r←a⍳b'r a b
→passed←0
rank_error:→~passed←1
unexpected_signal:#.failure_logger.recordFailure'Unexpected Signal'⎕DM'r←a⍳b'⍬ a b
→passed←0
```

Some Personal History 1969 - 1986

- Discovered APL/360 in 1969
- Used APL to teach Computing Courses
  - 1970 – 1973
  - Hampton, Virginia High Schools
- APL Applications and APL Implementations
  - 1974 – 1986
  - TCC, STSC, Burroughs, Data Resources, Analogic Corporation







Some Personal History 1986 - 2019

- Microsoft (1986-2007)
  - Windows, OS/2, NT, Visual Basic, Encarta, Microsoft Research, Time Travel Debugging, HTML Help, Full Text Search, Evangelism
- Zephyr Aviation (1998-2002)
- Podly.TV (2007-2012)
  - Internet Television
  - 8,000 channels for all subjects
    - Musical Artists, Special Interests, News

Some Personal History 1986-2019

- Amazon.com (2013-2015)
  - Unified Subledger for all Financial Transactions
    - Originally largely in Oracle Databases
    - Migrated large portions to DynamoDB on AWS
    - Continuous scaling to meet exponential growth
  - Test Driven Development
  - Continuous Refactoring

Some Personal History 1986-2019

- Microsoft Azure (2017-2019)
  - Archival Storage
    - Very low cost
    - Extremely high reliability
    - Very large data volumes
      - Terrabytes growing to Petabytes and Exabytes

Some Personal History – Time Travel Debugging

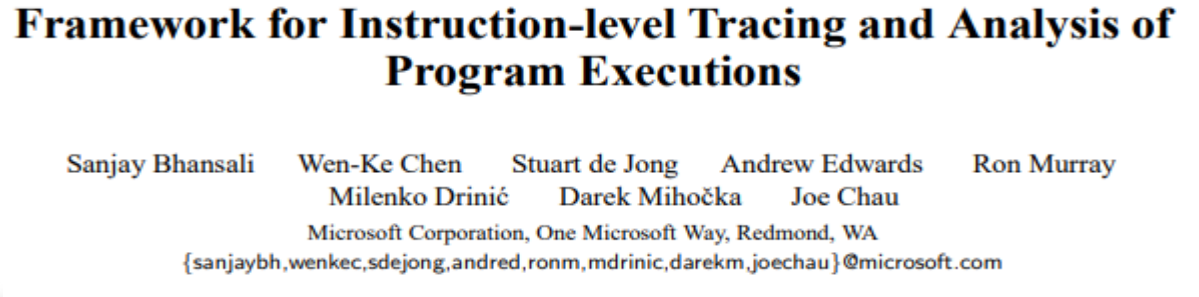





- TTD records every instruction executed
- Allows for replay in Windbg
  - Forward
  - Backwards
  - Supports all breakpoints (including Data!)
- Nails Heisenbugs!
  - Once recorded, bugs always replay exactly the same way!

Some Personal History – Time Travel Debugging

- References:
  - https://aka.ms/WinDbgPreview
  - https://www.usenix.org/legacy/events/vee06/full_papers/p154-bhansali.pdf
  - https://devblogs.microsoft.com/visualstudio/introducing-time-travel-debugging-for-visual-studio-enterprise-2019/



Framework for Instruction-level Tracing and Analysis of Program Executions

Sanjay Bhansali Wen-Ke Chen Stuart de Jong Andrew Edwards Ron Murray
Milenko Drinić Darek Mihočka Joe Chau

Microsoft Corporation, One Microsoft Way, Redmond, WA
{sanjaybh,wenkec,sdejong,andred,ronm,mdrinic,darekm,joechau}@microsoft.com

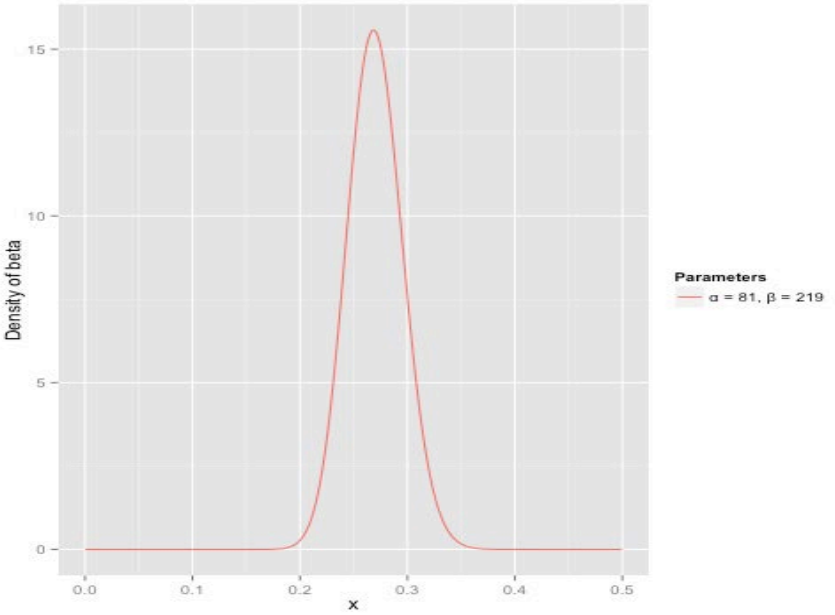
Additional Slides

Baseball and the Beta Distribution

- Simple Batting Average Statistics
- For the first batter of the first game
  - After One Hit
    - 100% batting average!
  - After One Strike-Out
    - 0% batting average!
- We can do better!
- See: http://varianceexplained.org/statistics/beta_distribution_and_baseball/

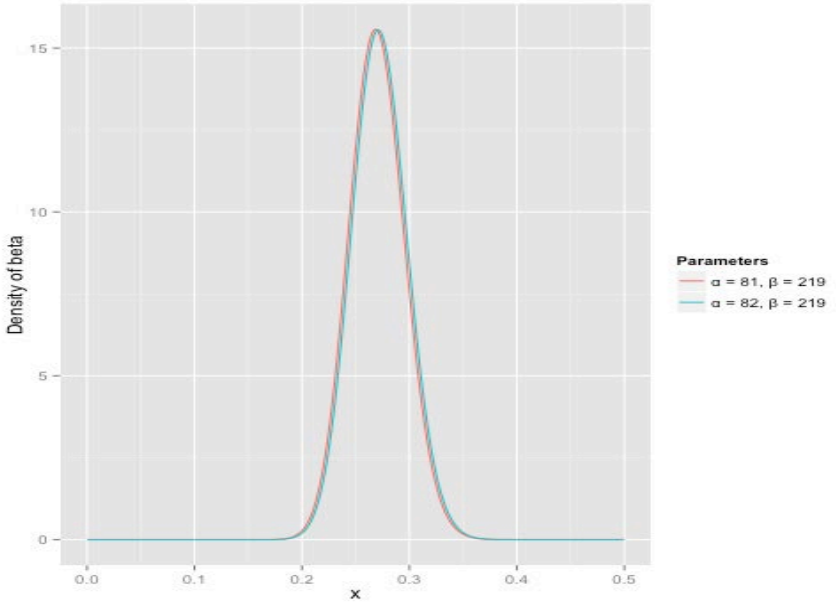
Baseball and the Beta Distribution

- Using the Beta Distribution
  - Used for Bayesian Reasoning
  - Parameters 81, 219
    - Expected value is 0.270
    - Average Batter: 0.266
    - Excellent Batter: 0.300



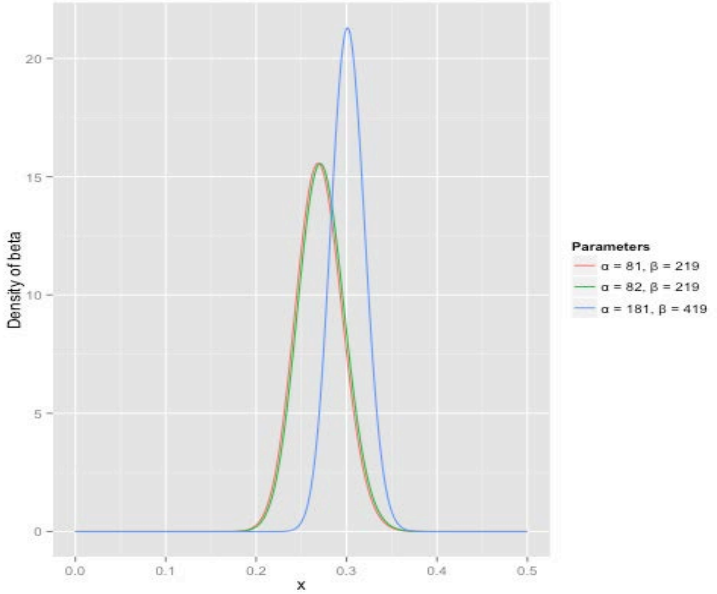
Baseball and the Beta Distribution

- If the batter gets one hit
  - New Beta Parameters
    - (81 219 + 1 0)
    - Changes things a tiny bit



Baseball and the Beta Distribution

- If after 300 at-bat tries
  - A batter gets 100 hits
    - Beta parameters (81 219 + 100 200)
    - We see a significant change

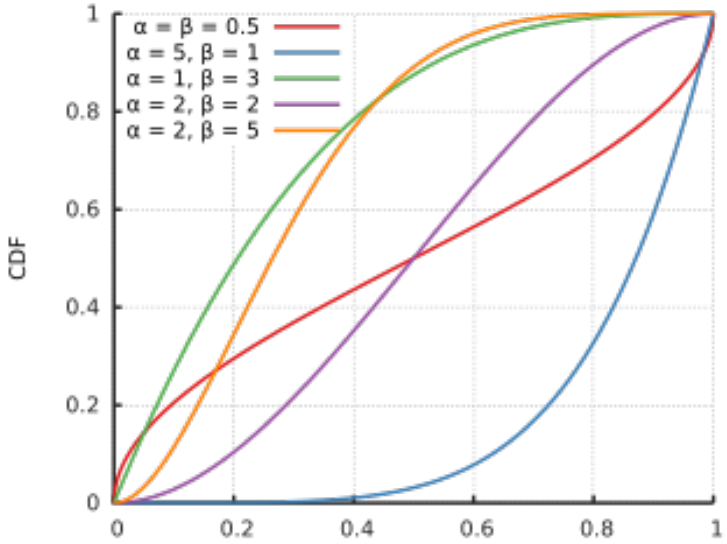


Current Strategies

- For Uniform Random Numbers
  - ?Nρ0
    - Uniform Distribution between 0 and 1
  - For numbers between Min and Max:
    - Min+(Max-Min)×?Nρ0
  - □RL
    - Selects algorithm
    - Sets Initial Seed Value

Current Strategies – Inverse Transform Sampling

- We use the Cumulative Probability Distribution



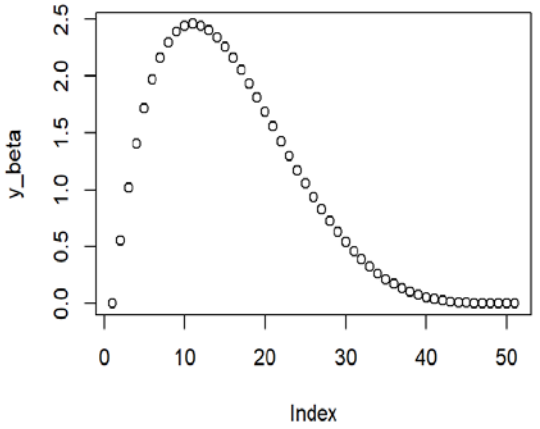
Current Strategies: Inverse Transform Sampling

```
:Namespace Beta
⎕IO←0

⍝ maxMesh←1000

Random←{inc←÷⍵ ⍝⌊maxMesh
intv←{0,+\⍵÷+/⍵}⍺ PDF(1+⍳⍵-1)×inc
urv←?⍵⍴0
i←intv⍸urv
inc×i+(urv-intv[i])÷intv[i+1]-intv[i]
}

PDF←{params←2⍴⍺ ⋄ a←params[0] ⋄ b←params[1] ⋄ ((!¯1+a+b)÷×/!params-1)×(⍵*a-1)×(1-⍵)*b-1}

:EndNamespace ⍝ Beta
```

https://en.wikipedia.org/wiki/Inverse_transform_sampling

Current Strategies: Using R

```
      )load rconnect
C:\Program Files\Dyalog\Dyalog APL-64 18.0 Unicode\ws\rconnect.dws saved
Wed Jul 15 20:03:50 2020
      r←⎕NEW R
      r.init
RConnect initialized
      r.x'x_beta<-seq(0, 1, by = 0.02)'
      r.x'y_beta<-dbeta(x_beta, shape1 = 2, shape2 = 5)'
      r.x'plot(y_beta)'

      samples←r.x'dbeta(x_beta, shape1 = 2, shape2 = 5)'
```



QA Work – Code Coverage

```
      )load coverage.dws
.\coverage.dws saved Tue Nov 3 11:01:46 2020
      Report Latest
```

Date	Time	File	Lines	Red	Excluded	% Covered
11/8/2020	13:10	grade.c	1168	0	27	100
11/8/2020	13:10	interval.cpp	831	74	0	91.09507
11/8/2020	13:10	membership.cpp	434	3	3	99.30876
11/8/2020	13:10	same.c	2302	37	16	98.3927
11/8/2020	13:10	unique.cpp	1213	26	21	97.85655

QA Work – Code Coverage

- Status on lines not covered:

```
      cLatest←⎕NEW Coverage Latest
      cLatest.StatisticsFor'membership'
2020-11-08 13:10 membership.cpp 434 3 3 99.30875576
      cLatest.RedLinesFor'membership'
2020-11-08 13:10 membership.cpp 353 431 483
```

QA Work – Writing Unit Tests as Documentation – Part 3

```
passed←r VerifyIntervalIndexResult arguments;a;b;LEQ;TestEachResult
(a b)←arguments ⋄ leq_result←⎕IO+0 1
LEQ←{leq_result≡α{⍲(⊂α),⊂ω}ω}
TestEachResult←{
bottom←⎕IO ⋄ top←⎕IO+¯1+1↑⍴a
α<⎕IO:(ω⌷b)LEQ(⎕IO⌷a)
(α≥top):(top⌷a)LEQ(ω⌷b)
((α⌷a)LEQ(ω⌷b))∧(ω⌷b)LEQ((α+1)⌷a)
}
passed←∧/r TestEachResult¨⍳⍴r
```