



# The Rayon Visualization Toolkit

**Phil Groce**  
**CERT Network Situational  
Awareness Group (NetSA)**



---

© 2010 Carnegie Mellon University

## NO WARRANTY

THIS MATERIAL OF CARNEGIE MELLON UNIVERSITY AND ITS SOFTWARE ENGINEERING INSTITUTE IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

This presentation may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at [permission@sei.cmu.edu](mailto:permission@sei.cmu.edu).

This work was created in the performance of Federal Government Contract Number FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center. The government of the United States has a royalty-free government-purpose license to use, duplicate, or disclose the work, in whole or in part and in any manner, and to have or permit others to do so, for government purposes pursuant to the copyright license under the clause at 252.227-7013.

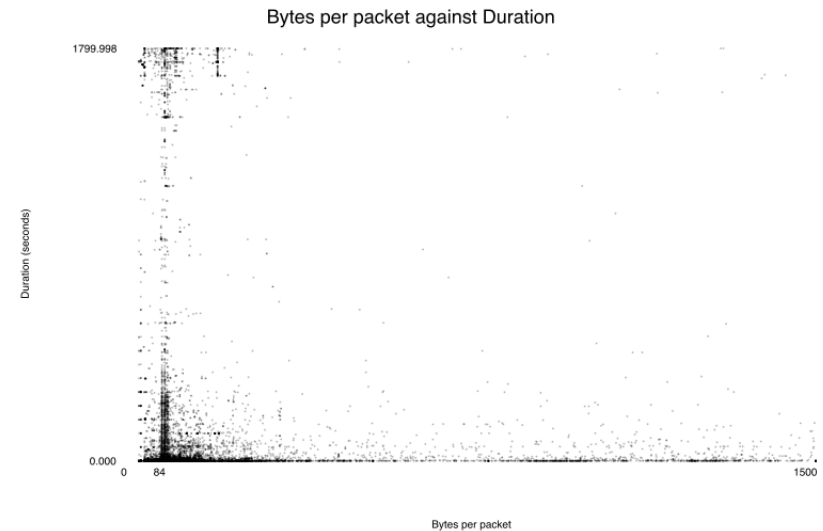
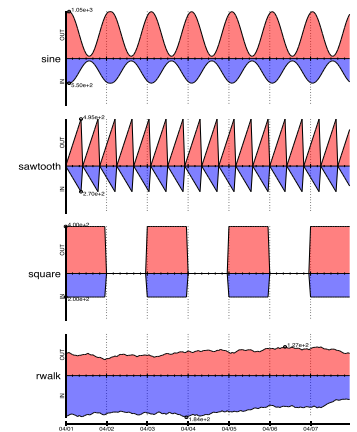
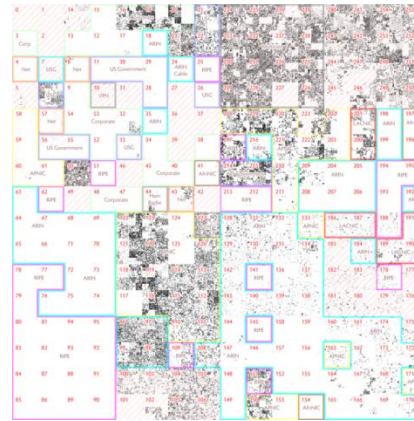
CERT® is a registered mark owned by Carnegie Mellon University.

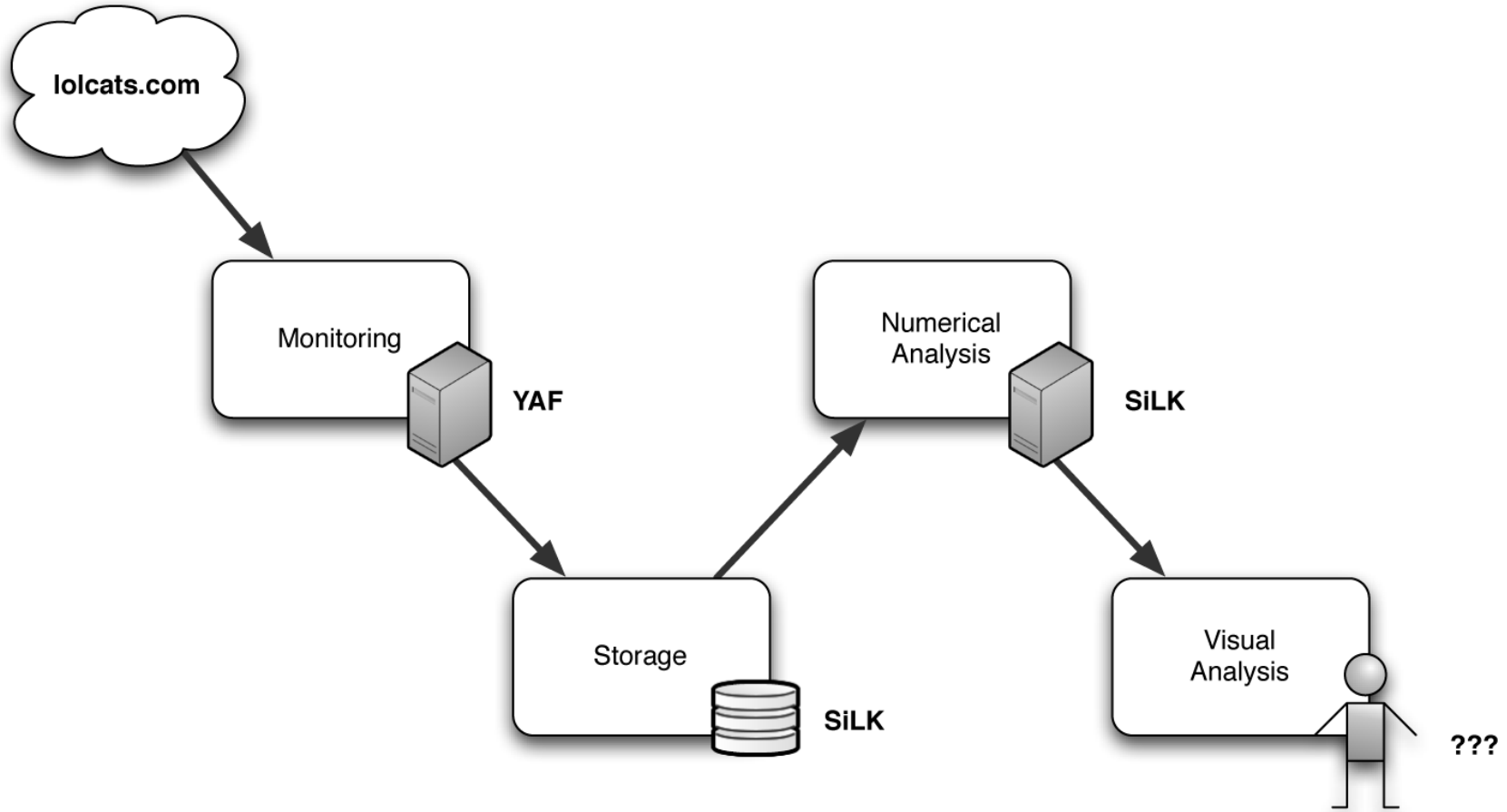
# Motivation

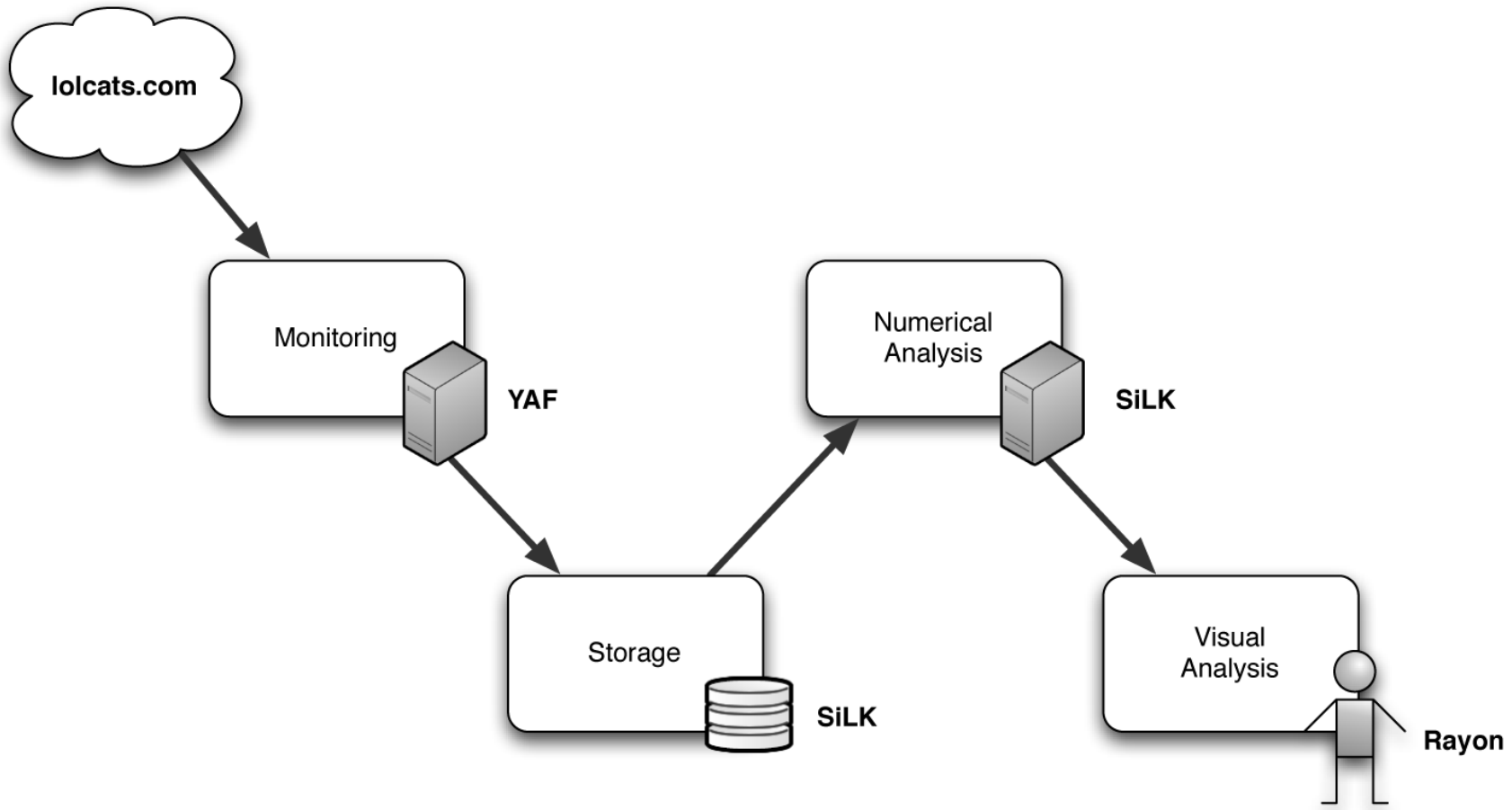
Improve transition/uptake of NetSA analytics

Provide basic visualization that people in SOCs can use easily

- Live where they live



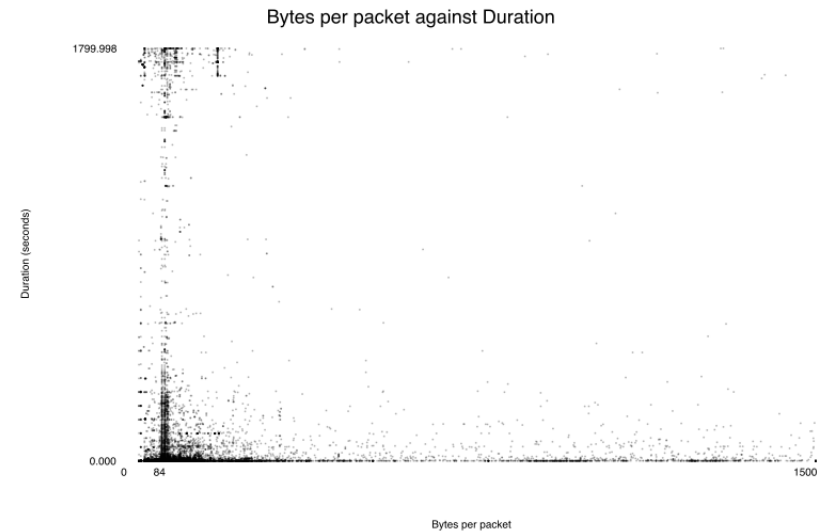
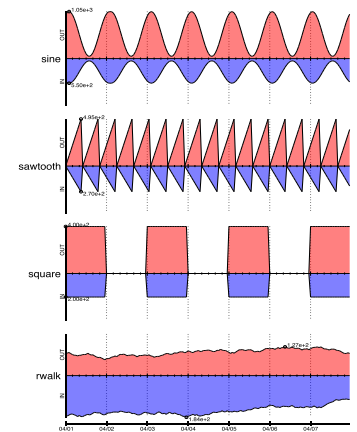




# Motivation

## Visualize SiLK data

- Live where SiLK lives (Unix, command-line)
- Live in iSiLK



# Rayon Fun Facts™

---

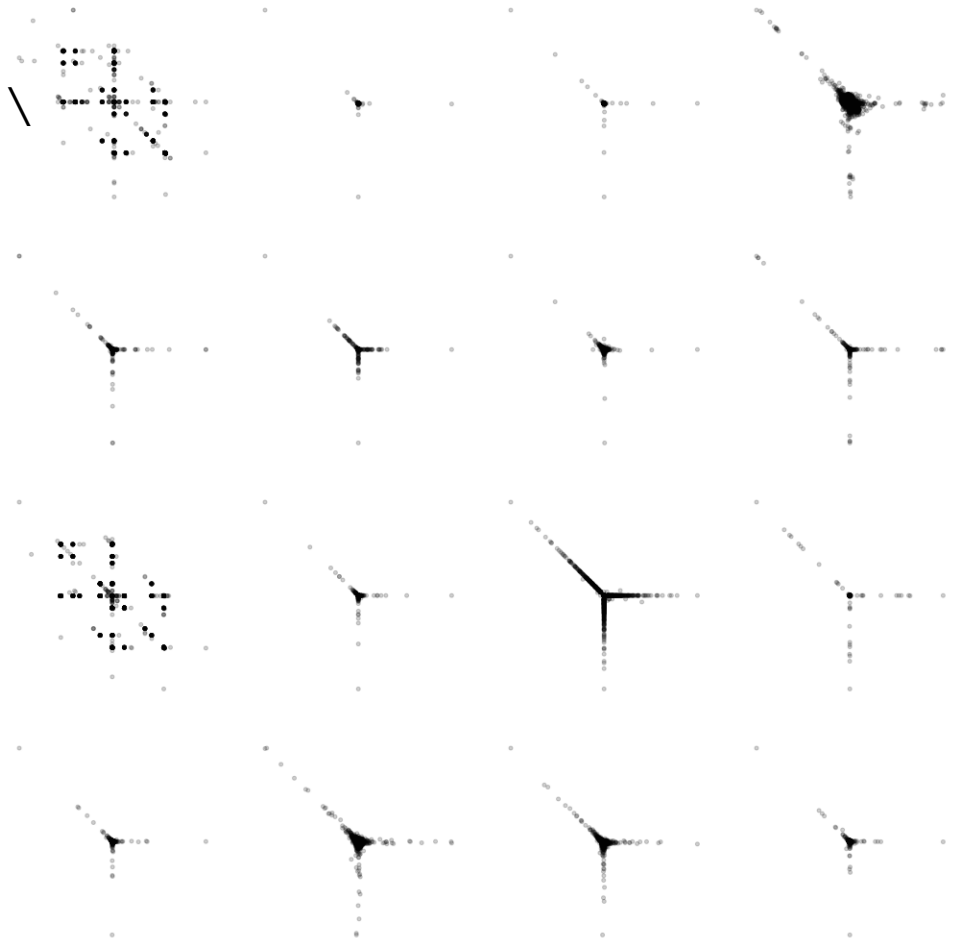
- Can render visualizations to:
  - PDF, SVG, PNG (via Cairo)
  - GUI (via wxPython)
- Requirements:
  - Python >2.4, < 3.0
  - One or both of
    - Cairo and PyCairo (1.4.x and 1.8.x tested)
    - wxWidgets and wxPython (2.8.x tested)

# ryscatterplot

---

```
rysccatterplot --input-path=foo.txt \  
--output-path=foo.svg \  
--x-input=1 --y-input=2 \  
--grid --grid-key-input=0
```

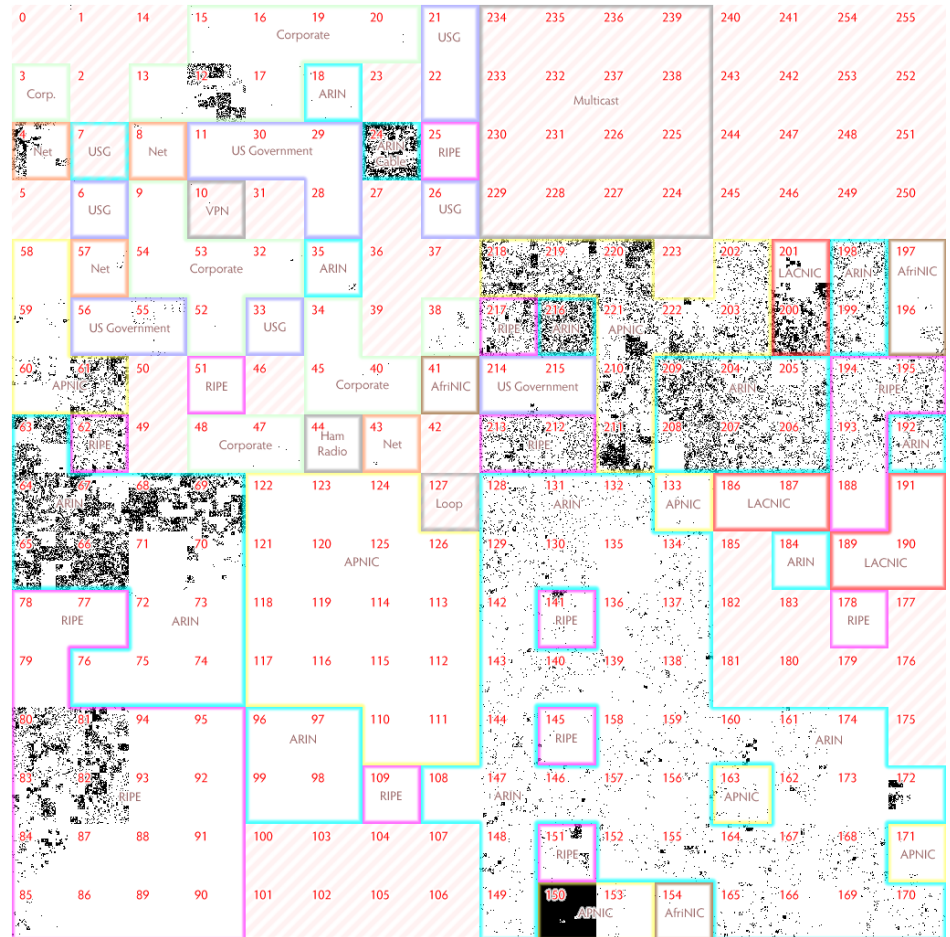
```
## key | x | y  
1.2.3.4 | 0.0 | 0.0  
1.2.3.4 | 0.0 | 200.0  
...  
5.6.7.8 | 144.0 | 0.0
```





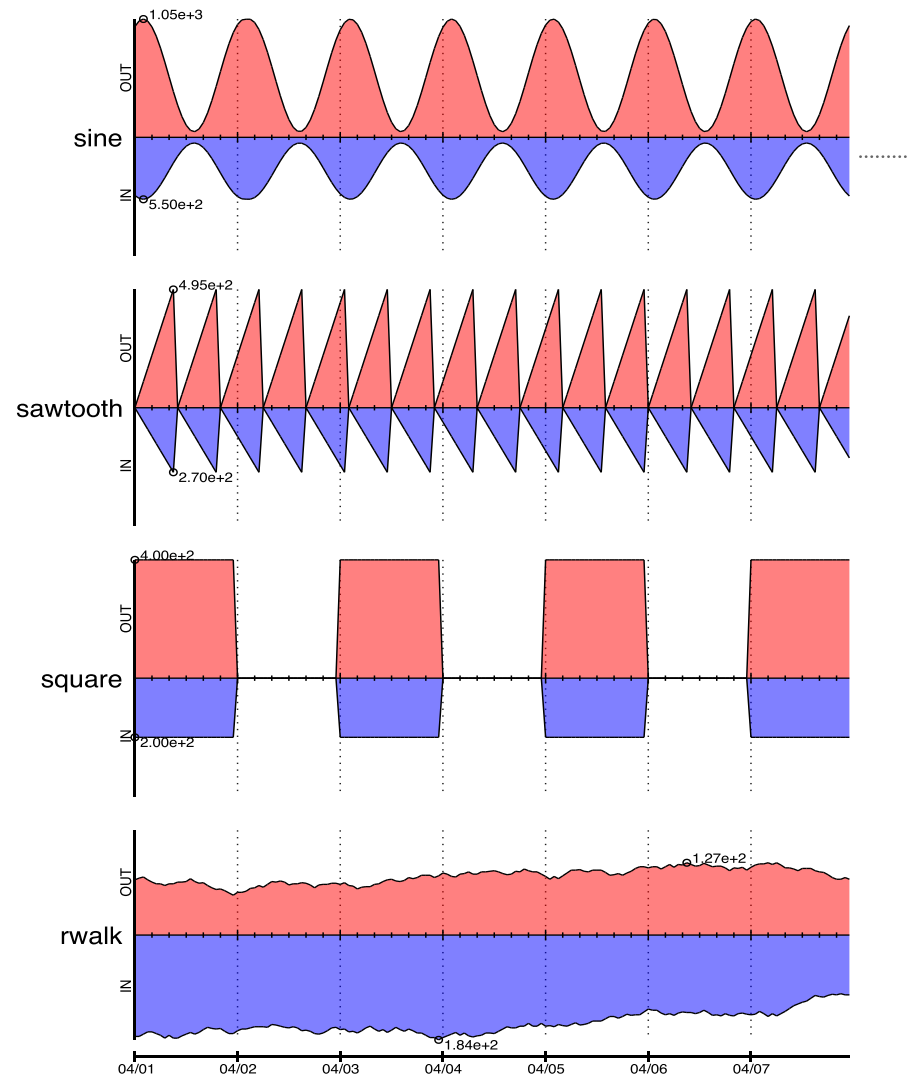
# ryhilbert

```
rwsetcat foo.set | \  
ryhilbert --input-path - --output-path foo.png \  
--binary-plot
```



# rystripplot

```
rystripplot \
--in foo.txt \
--out bar.png
```

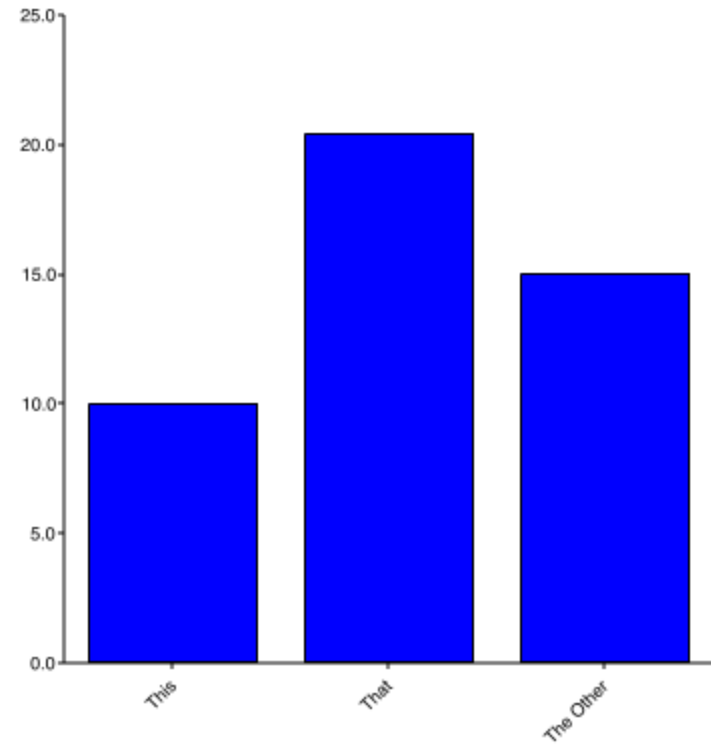


```
## date|sine_in|sine_out|sawtooth_in|sawtooth_out|square_in|square_out|rwalk_in|rwalk_out
2000-04-01 00:00:00+00:00|982.74|516.37|0.00|0.00|400.00|200.00|97.00|178.00
2000-04-01 01:00:00+00:00|1033.26|541.63|55.00|30.00|400.00|200.00|100.00|178.00
2000-04-01 02:00:00+00:00|1049.99|550.00|110.00|60.00|400.00|200.00|102.00|174.00
2000-04-01 03:00:00+00:00|1031.77|540.88|165.00|90.00|400.00|200.00|97.00|170.00
```

# rycategories

---

```
rycategories \  
--in foo.txt \  
--out bar.png
```

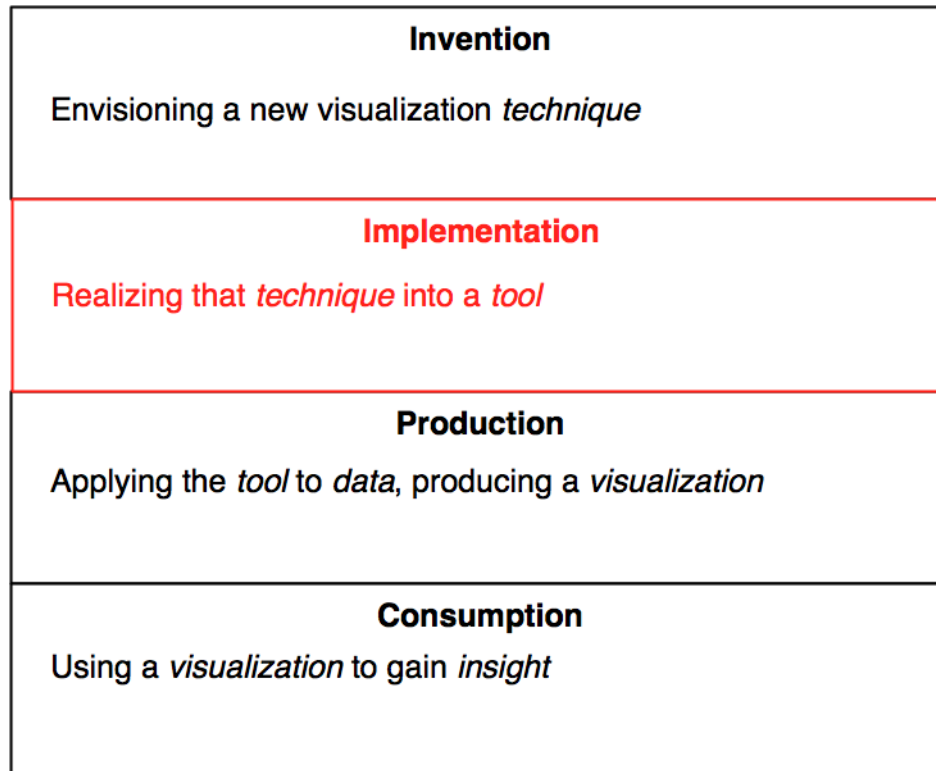


# Phases of Visualization

---

<b>Invention</b> Envisioning a new visualization <i>technique</i>
<b>Implementation</b> Realizing that <i>technique</i> into a <i>tool</i>
<b>Production</b> Applying the <i>tool</i> to <i>data</i> , producing a <i>visualization</i>
<b>Consumption</b> Using a <i>visualization</i> to gain <i>insight</i>





# Code Sample

```
from rayon import toolbox

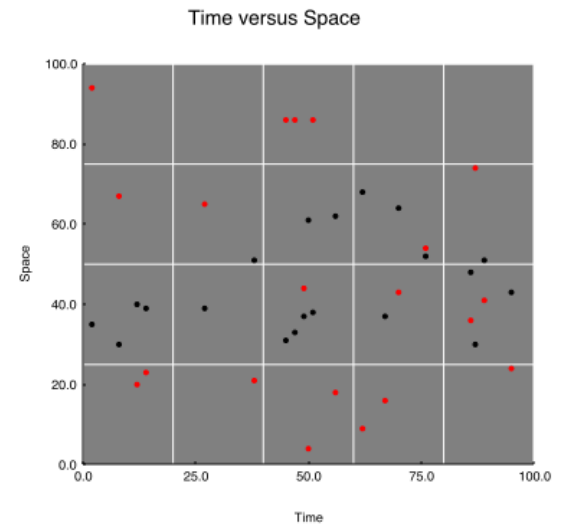
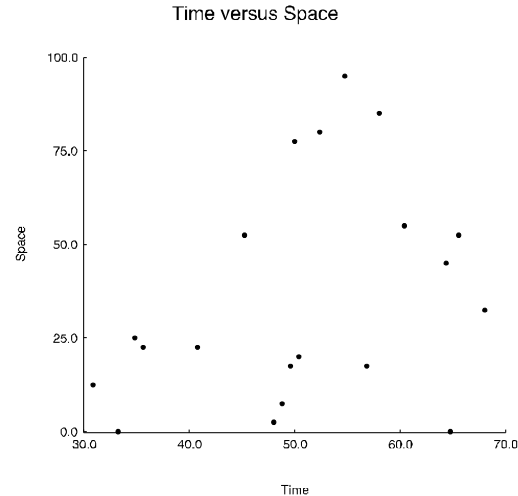
tools = toolbox.Toolbox.for_file()

# Read in data
indata = toolbox.new_dataset_from_filename(
    "sample_in.txt")

# Define the chart
chart = tools.new_chart("square")
plt = tools.new_plot("scatter")
plt.set_data(x=indata.column(0),
             y=indata.column(1))
chart.add_plot(plt)
c.set_chart_background("white")

# Decorate chart - http://tools.netsa.cert.org
# for more
omitted_for_space()

# Draw the chart
page = tools.new_page_from_filename(
    outfile, width=400, height=400)
page.write(chart)
```



# Importing and Manipulating Data

---

```
## Typemap: str,int,str,int
## proto|port|network|count
TCP|8080|A|1009
UDP|8080|A|1001388
TCP|25|A|4396
TCP|53|B|230
UDP|25|A|4
...
```

```
from rayon.data import *
d = Dataset.from_file('foo.txt')
c = d.get_column('proto')
c2 = column([1,2,1,2,3,...])
d.add_column(c2,
             name="stuff")

d2 = d.map(lambda r:
           [r.proto, r.count+stuff])

d.to_file('bar.txt')
D2.to_file('baz.txt')
```

# Extending Rayon

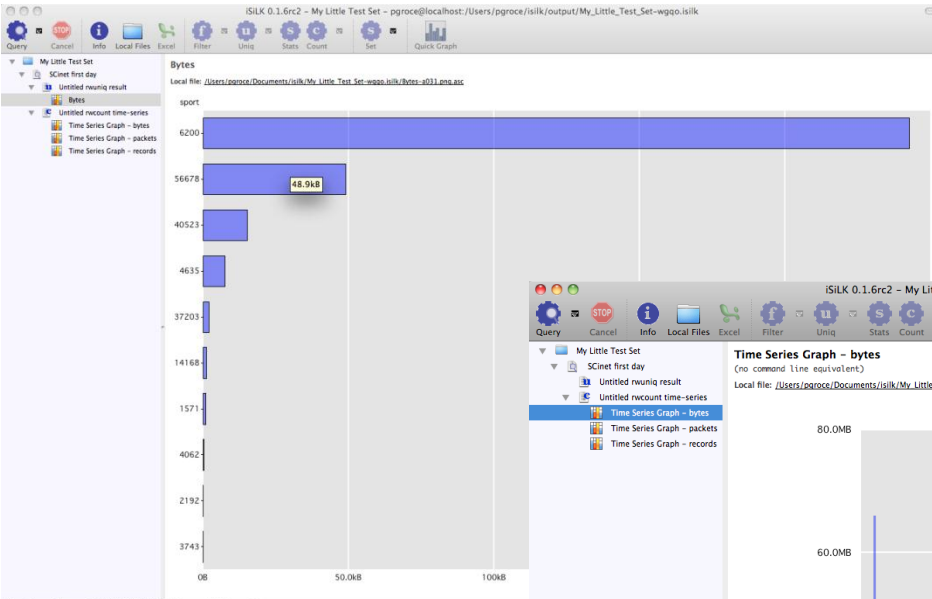
---

```
import math
from rayon.plots import *
from rayon.markers import *

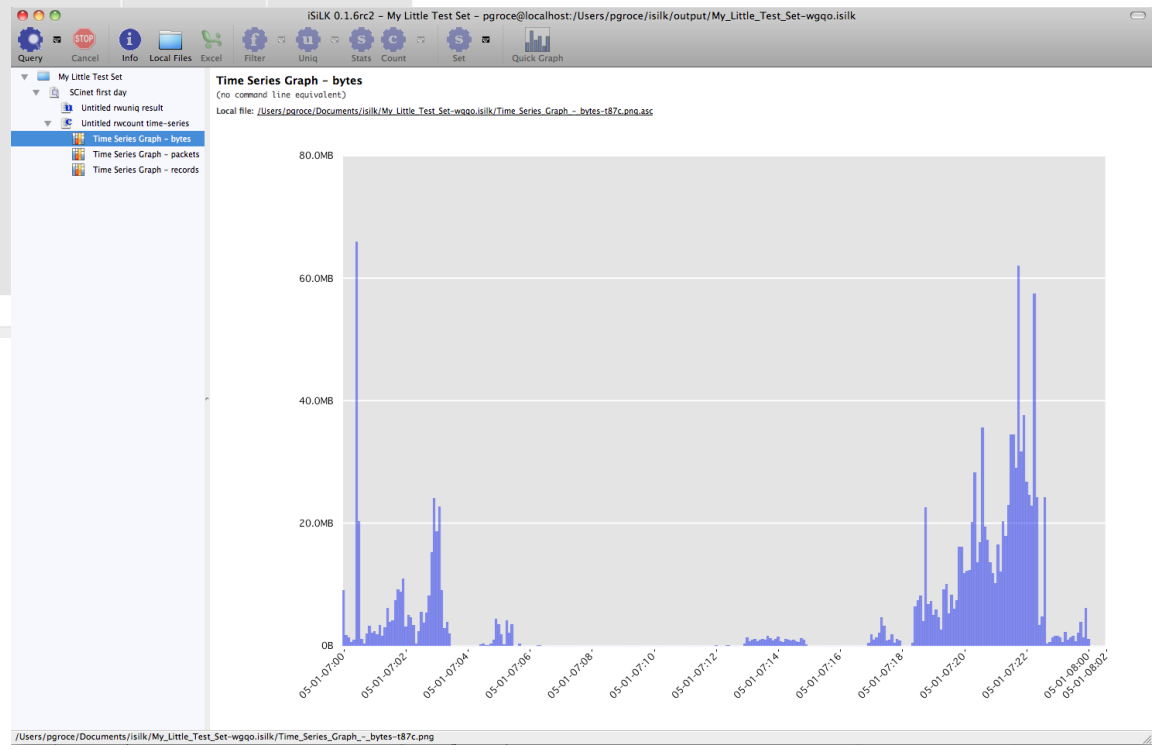
class PolarScatterPlot(plots.Plot):
    axes = ('r', 'theta')
    def draw_(self, ctx, width, height):
        marker = markers.Dot()
        for r, theta in self.get_scaled_points():
            x = r * math.cos(theta)
            y = r * math.sin(theta)
            marker.draw(ctx,
                       x * width,
                       height - (y * height))
```



# Rayon and iSiLK



/Users/pgroce/Documents/isilk/My\_Little\_Test\_Set-wgqo.isilk/Bytes-a031.png



/Users/pgroce/Documents/isilk/My\_Little\_Test\_Set-wgqo.isilk/Time\_Series\_Graph\_-\_bytes-187c.png

# Rayon Status

---

Current Version: 1.0.1

- Released 2010.11.10
- <http://tools.netsa.cert.org/rayon>

Questions?

