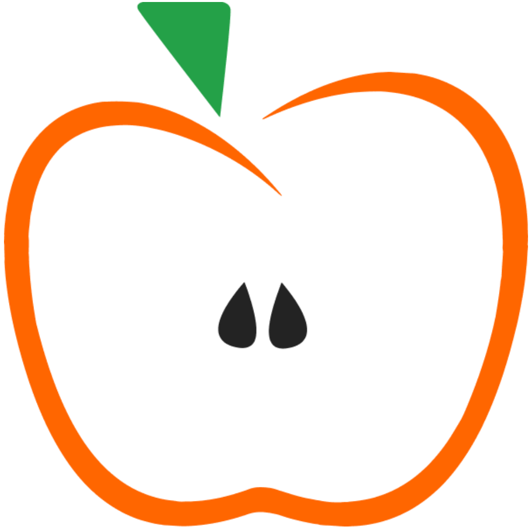
DYALOG

APL Seeds '24

What is APL
and
What Can APL Do For You?



Adám Brudzewsky

Myth: “APL is Unreadable”

$(\times/!x-1)\div!(+/x)-1$

$$\frac{\prod_{i=1}^{n}(x_i-1)!}{\left(\left(\sum_{i=1}^{n}x_i\right)-1\right)!}$$

ฟังก์ชันเบต้าหลายตัวแปร

دالة بيتا متعددة المتغيرات

多元贝塔函数

Myth: “APL is Unreadable”

```
x.map(e=>[...Array(e).keys()].slice(1).reduce((a,b)=>a*b,1)).reduce((a,b)=>a*b)
  /[...Array(x.reduce((a,b)=>a+b)).keys()].slice(1).reduce((a,b)=>a*b,1)
```

Myth: “APL is Unreadable”

```
x.map(e=>[...Array(e).keys()].slice(1).reduce((a,b)=>a*b,1)).reduce((a,b)=>a*b)
 /[...Array(x.reduce((a,b)=>a+b)).keys()].slice(1).reduce((a,b)=>a*b,1)
```

```
Fac=x=>[...Array(x+1).keys()].slice(1).reduce((a,b)=>a*b,1)
x.map(e=>Fac(e-1)).reduce((a,b)=>a*b)/Fac(x.reduce((a,b)=>a+b)-1)
```

Myth: “APL is Unreadable”

```
x.map(e=>[...Array(e).keys()].slice(1).reduce((a,b)=>a*b,1)).reduce((a,b)=>a*b)
  /[...Array(x.reduce((a,b)=>a+b)).keys()].slice(1).reduce((a,b)=>a*b,1)
```

```
Fac=x=>[...Array(x+1).keys()].slice(1).reduce((a,b)=>a*b,1)
x.map(e=>Fac(e-1)).reduce((a,b)=>a*b)/Fac(x.reduce((a,b)=>a+b)-1)
```

```
Sum=x=>x.reduce((a,b)=>a+b,0)
Prd=x=>x.reduce((a,b)=>a*b,1)
Rng=x=>[...Array(x).keys()]
Fac=x=>Prd(Rng(x+1).slice(1))
Prd(x.map(e=>Fac(e-1)))/Fac(Sum(x)-1)
```

Myth: “APL is Unreadable”

```
x.map(e=>[...Array(e).keys()].slice(1).reduce((a,b)=>a*b,1)).reduce((a,b)=>a*b)
  /[...Array(x.reduce((a,b)=>a+b)).keys()].slice(1).reduce((a,b)=>a*b,1)

Fac=x=>[...Array(x+1).keys()].slice(1).reduce((a,b)=>a*b,1)
x.map(e=>Fac(e-1)).reduce((a,b)=>a*b)/Fac(x.reduce((a,b)=>a+b)-1)
```

```
Sum = x => x.reduce((a, b) => a + b, 0)
Prd = x => x.reduce((a, b) => a * b, 1)
Rng = x => [... Array(x).keys()]
Fac = x => Prd(Rng(x + 1).slice(1))
Prd(x.map(e => Fac(e - 1))) / Fac(Sum(x) - 1)
```

Myth: “APL is Unreadable”

```
x.map(e=>[...Array(e).keys()].slice(1).reduce((a,b)=>a*b,1)).reduce((a,b)=>a*b)
  /[...Array(x.reduce((a,b)=>a+b)).keys()].slice(1).reduce((a,b)=>a*b,1)
```

```
Fac=x=>[...Array(x+1).keys()].slice(1).reduce((a,b)=>a*b,1)
x.map(e=>Fac(e-1)).reduce((a,b)=>a*b)/Fac(x.reduce((a,b)=>a+b)-1)
```

```
Sum = x => x.reduce((a, b) => a + b, 0)          Sum ← +/
Prd = x => x.reduce((a, b) => a * b, 1)          Prd ← ×/
Rng = x => [... Array(x).keys()]                 Rng ← ⍳
Fac = x => Prd(Rng(x + 1).slice(1))              Fac ← Prd 1 ↓ (Rng +∘1)
Prd(x.map(e => Fac(e - 1))) / Fac(Sum(x) - 1)   (Prd Fac¨x - 1) ÷ Fac(Sum x) - 1
```

Myth: “APL is Unreadable”

```
x.map(e=>[...Array(e).keys()].slice(1).reduce((a,b)=>a*b,1)).reduce((a,b)=>a*b)
  /[...Array(x.reduce((a,b)=>a+b)).keys()].slice(1).reduce((a,b)=>a*b,1)
```

```
Fac=x=>[...Array(x+1).keys()].slice(1).reduce((a,b)=>a*b,1)
x.map(e=>Fac(e-1)).reduce((a,b)=>a*b)/Fac(x.reduce((a,b)=>a+b)-1)
```

```
Sum = x => x.reduce((a, b) => a + b, 0)
Prd = x => x.reduce((a, b) => a * b, 1)
Rng = x => [... Array(x).keys()]
Fac = x => Prd(Rng(x + 1).slice(1))
Prd(x.map(e => Fac(e - 1))) / Fac(Sum(x) - 1)
```

```
Sum ← +/
Prd ← ×/
Rng ← ⍳
Fac ← Prd 1 ↓ (Rng +∘1)
(Prd Fac¨x - 1) ÷ Fac(Sum x) - 1
```

Myth: “APL is Unreadable”

```
x.map(e=>[...Array(e).keys()].slice(1).reduce((a,b)=>a*b,1)).reduce((a,b)=>a*b)
  /[...Array(x.reduce((a,b)=>a+b)).keys()].slice(1).reduce((a,b)=>a*b,1)
```

```
Fac=x=>[...Array(x+1).keys()].slice(1).reduce((a,b)=>a*b,1)
x.map(e=>Fac(e-1)).reduce((a,b)=>a*b)/Fac(x.reduce((a,b)=>a+b)-1)
```

```
Sum = x => x.reduce((a, b) => a + b, 0)
Prd = x => x.reduce((a, b) => a * b, 1)
Rng = x => [... Array(x).keys()]
Fac = x => Prd(Rng(x + 1).slice(1))
Prd(x.map(e => Fac(e - 1))) / Fac(Sum(x) - 1)
```

```
Sum ← +/
Prd ← ×/
Rng ← ⍳
Fac ← Prd 1 ↓ (Rng +∘1)
(Prd Fac¨x - 1) ÷ Fac(Sum x) - 1
```

Myth: “APL is Unreadable”

```
x.map(e=>[...Array(e).keys()].slice(1).reduce((a,b)=>a*b,1)).reduce((a,b)=>a*b)
  /[...Array(x.reduce((a,b)=>a+b)).keys()].slice(1).reduce((a,b)=>a*b,1)

Fac=x=>[...Array(x+1).keys()].slice(1).reduce((a,b)=>a*b,1)
x.map(e=>Fac(e-1)).reduce((a,b)=>a*b)/Fac(x.reduce((a,b)=>a+b)-1)

Sum = x => x.reduce((a, b) => a + b, 0)          Sum ← +/
Prd = x => x.reduce((a, b) => a * b, 1)          Prd ← ×/
Rng = x => [... Array(x).keys()]                 Rng ← ⍳
Fac = x => Prd(Rng(x + 1).slice(1))              Fac ← Prd 1 ↓ (Rng +∘1)
Prd(x.map(e => Fac(e - 1))) / Fac(Sum(x) - 1)    (Prd Fac¨x - 1) ÷ Fac(Sum x) - 1
```

Myth: “APL is Unreadable”

```
x.map(e=>[...Array(e).keys()].slice(1).reduce((a,b)=>a*b,1)).reduce((a,b)=>a*b)
  /[...Array(x.reduce((a,b)=>a+b)).keys()].slice(1).reduce((a,b)=>a*b,1)
```

```
Fac=x=>[...Array(x+1).keys()].slice(1).reduce((a,b)=>a*b,1)
x.map(e=>Fac(e-1)).reduce((a,b)=>a*b)/Fac(x.reduce((a,b)=>a+b)-1)
```

```
Sum = x => x.reduce((a, b) => a + b, 0)
Prd = x => x.reduce((a, b) => a * b, 1)
Rng = x => [... Array(x).keys()]
Fac = x => Prd(Rng(x + 1).slice(1))
Prd(x.map(e => Fac(e - 1))) / Fac(Sum(x) - 1)
```

```
Sum ← +/
Prd ← ×/
Rng ← ⍳
Fac ← Prd 1 ↓ (Rng +∘1)
(Prd Fac¨x - 1) ÷ Fac(Sum x) - 1
```

Myth: “APL is Unreadable”

```
x.map(e=>[...Array(e).keys()].slice(1).reduce((a,b)=>a*b,1)).reduce((a,b)=>a*b)
  /[...Array(x.reduce((a,b)=>a+b)).keys()].slice(1).reduce((a,b)=>a*b,1)
```

```
Fac=x=>[...Array(x+1).keys()].slice(1).reduce((a,b)=>a*b,1)
x.map(e=>Fac(e-1)).reduce((a,b)=>a*b)/Fac(x.reduce((a,b)=>a+b)-1)
```

```
Sum = x => x.reduce((a, b) => a + b, 0)        Sum ← +/
Prd = x => x.reduce((a, b) => a * b, 1)        Prd ← ×/
Rng = x => [... Array(x).keys()]               Rng ← ⍳
Fac = x => Prd(Rng(x + 1).slice(1))            Fac ← Prd 1 ↓ (Rng +∘1)
Prd(x.map(e => Fac(e - 1))) / Fac(Sum(x) - 1)  (Prd Fac¨x - 1) ÷ Fac(Sum x) - 1
```

Myth: “APL is Unreadable”

```
x.map(e=>[...Array(e).keys()].slice(1).reduce((a,b)=>a*b,1)).reduce((a,b)=>a*b)
  /[...Array(x.reduce((a,b)=>a+b)).keys()].slice(1).reduce((a,b)=>a*b,1)
```

```
Fac=x=>[...Array(x+1).keys()].slice(1).reduce((a,b)=>a*b,1)
x.map(e=>Fac(e-1)).reduce((a,b)=>a*b)/Fac(x.reduce((a,b)=>a+b)-1)
```

```
Sum = x => x.reduce((a, b) => a + b, 0)       Sum ← +/
Prd = x => x.reduce((a, b) => a * b, 1)       Prd ← ×/
Rng = x => [... Array(x).keys()]              Rng ← ⍳
Fac = x => Prd(Rng(x + 1).slice(1))           Fac ← Prd 1 ↓ (Rng +∘1)
Prd(x.map(e => Fac(e - 1))) / Fac(Sum(x) - 1) (Prd Fac x - 1) ÷ Fac(Sum x) - 1
```

Myth: “APL is Unreadable”

```
x.map(e=>[...Array(e).keys()].slice(1).reduce((a,b)=>a*b,1)).reduce((a,b)=>a*b)
  /[...Array(x.reduce((a,b)=>a+b)).keys()].slice(1).reduce((a,b)=>a*b,1)

Fac=x=>[...Array(x+1).keys()].slice(1).reduce((a,b)=>a*b,1)
x.map(e=>Fac(e-1)).reduce((a,b)=>a*b)/Fac(x.reduce((a,b)=>a+b)-1)

Sum = x => x.reduce((a, b) => a + b, 0)        Sum ← +/
Prd = x => x.reduce((a, b) => a * b, 1)        Prd ← ×/
Rng = x => [... Array(x).keys()]
Fac = x => Prd(Rng(x + 1).slice(1))            Fac ← !
Prd(x.map(e => Fac(e - 1))) / Fac(Sum(x) - 1)  (Prd Fac x - 1) ÷ Fac(Sum x) - 1
```

Myth: “APL is Unreadable”

```
x.map(e=>[...Array(e).keys()].slice(1).reduce((a,b)=>a*b,1)).reduce((a,b)=>a*b)
 /[...Array(x.reduce((a,b)=>a+b)).keys()].slice(1).reduce((a,b)=>a*b,1)
```

```
Fac=x=>[...Array(x+1).keys()].slice(1).reduce((a,b)=>a*b,1)
x.map(e=>Fac(e-1)).reduce((a,b)=>a*b)/Fac(x.reduce((a,b)=>a+b)-1)
```

```
Sum = x => x.reduce((a, b) => a + b, 0)        Sum ← +/
Prd = x => x.reduce((a, b) => a * b, 1)        Prd ← ×/
Rng = x => [... Array(x).keys()]
Fac = x => Prd(Rng(x + 1).slice(1))            Fac ← !
Prd(x.map(e => Fac(e - 1))) / Fac(Sum(x) - 1)  (Prd Fac x - 1) ÷ Fac(Sum x) - 1
```

Myth: “APL is Unreadable”

```
x.map(e=>[...Array(e).keys()].slice(1).reduce((a,b)=>a*b,1)).reduce((a,b)=>a*b)
  /[...Array(x.reduce((a,b)=>a+b)).keys()].slice(1).reduce((a,b)=>a*b,1)

Fac=x=>[...Array(x+1).keys()].slice(1).reduce((a,b)=>a*b,1)
x.map(e=>Fac(e-1)).reduce((a,b)=>a*b)/Fac(x.reduce((a,b)=>a+b)-1)

Sum = x => x.reduce((a, b) => a + b, 0)
Prd = x => x.reduce((a, b) => a * b, 1)
Rng = x => [... Array(x).keys()]
Fac = x => Prd(Rng(x + 1).slice(1))
Prd(x.map(e => Fac(e - 1))) / Fac(Sum(x) - 1)  ( ×/  !  x - 1) ÷  ! ( +/ x) - 1
```

Myth: “APL is Unreadable”

```
x.map(e=>[...Array(e).keys()].slice(1).reduce((a,b)=>a*b,1)).reduce((a,b)=>a*b)
  /[...Array(x.reduce((a,b)=>a+b)).keys()].slice(1).reduce((a,b)=>a*b,1)
```

$$(\times/!x-1)\div!(+/x)-1$$

```
Fac=x=>[...Array(x+1).keys()].slice(1).reduce((a,b)=>a*b,1)
x.map(e=>Fac(e-1)).reduce((a,b)=>a*b)/Fac(x.reduce((a,b)=>a+b)-1)
```

```
Sum = x => x.reduce((a, b) => a + b, 0)
Prd = x => x.reduce((a, b) => a * b, 1)
Rng = x => [... Array(x).keys()]
Fac = x => Prd(Rng(x + 1).slice(1))
Prd(x.map(e => Fac(e - 1))) / Fac(Sum(x) - 1)  (×/!x-1)÷!(+/x)-1
```

What is APL?

~~Un~~readable

Array-oriented Programming Language

What is APL?

Symbolic

Array-oriented Programming Language

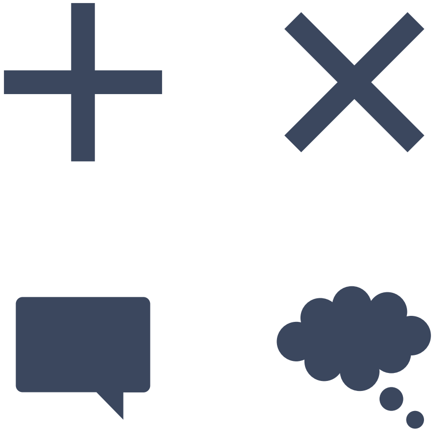
for Communicating Algorithms

to Computers

and Humans

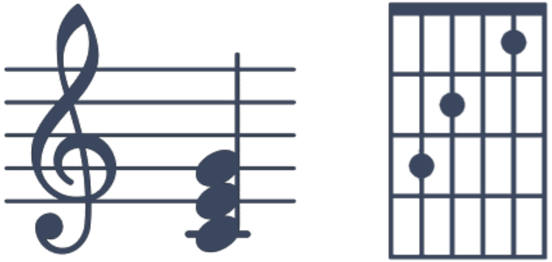
What is APL?

Symbolic



`slice()`

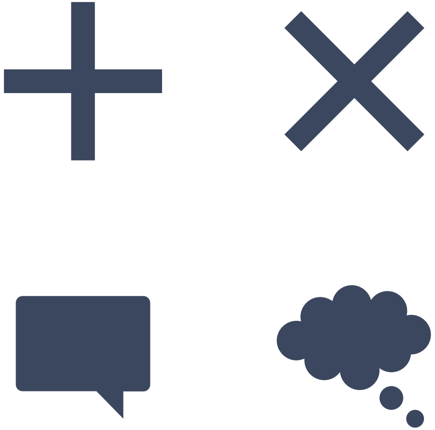
↓



网 田

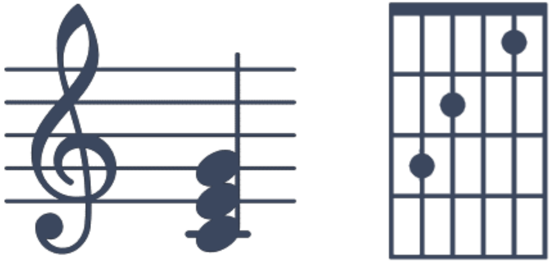
What is APL?

Symbolic



`slice()`

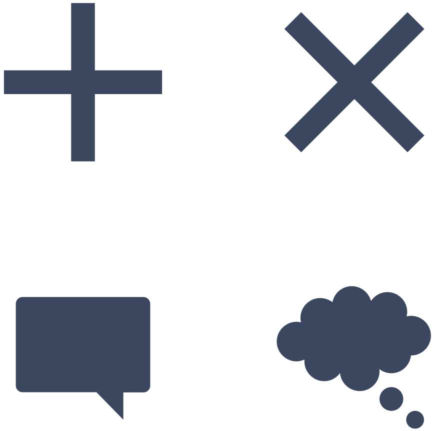
↓



网 田

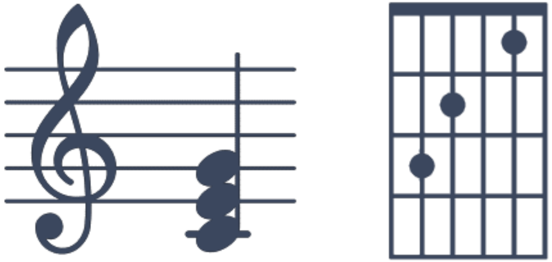
What is APL?

Symbolic



`slice()`

↓



网 田

What is APL?

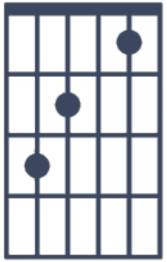
Symbolic

+ ×

💬 💭

↓ ↑

⌽ ⍉



网 田

What is APL?

Symbolic

Array-oriented Programming Language

```
(5,6,7,8)
```

What is APL?

Symbolic

Array-oriented Programming Language

```
      2+(5,6,7,8)
7 8 9 10
```

What is APL?

Symbolic

Array-oriented Programming Language

```
2↓(5,6,7,8)
```

```
7 8
```

What is APL?

Symbolic

Array-oriented Programming Language

```
      2↓(5,6,7,8)
7 8
```

What is APL?

Symbolic

Array-oriented Programming Language

```
2↑(5,6,7,8)
```

```
5 6
```

What is APL?

Symbolic

Array-oriented Programming Language

```
⌽(5,6,7,8)
8 7 6 5
```

What is APL?

Symbolic

Array-oriented Programming Language

table

T	r	y
A	P	L
n	o	w

⌽table

y	r	T
L	P	A
w	o	n

What is APL?

Symbolic

Array-oriented Programming Language

```
table
```

T	r	y
A	P	L
n	o	w

```
2↓⌽table
```

w	o	n

What is APL?

Symbolic

Array-oriented Programming Language

```
table
```

T	r	y
A	P	L
n	o	w

```
2 1↓⌽table
```

o	n

What is APL?

Symbolic

Array-oriented Programming Language

```
table
```

T	r	y
A	P	L
n	o	w

```
⍉2 1↓⌽table
```

o
n

What is APL?

What Can APL Do For You?

Help You

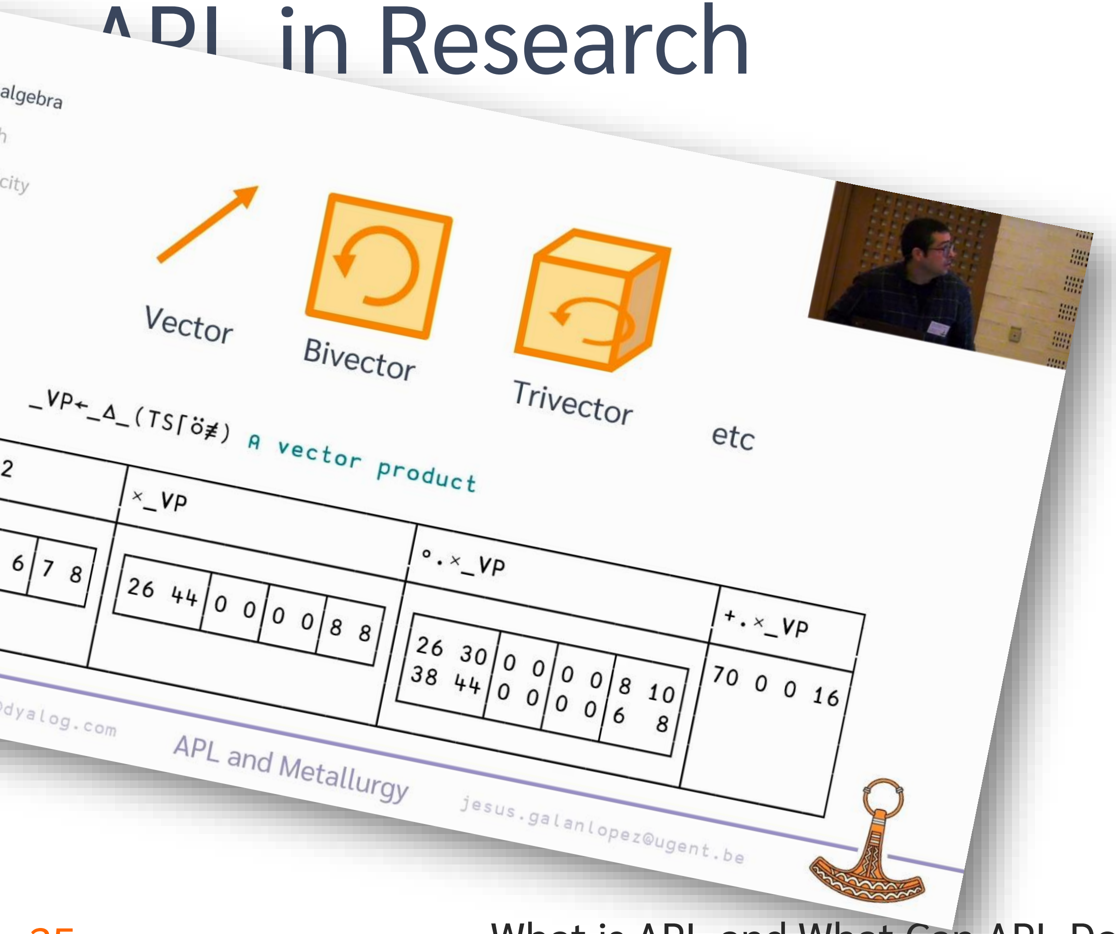
Communicate Algorithms

to Computers

and Humans

in Research, in Academia, in Industry, and More…

APL in Research



algebra

h

city

```
_VP←_∆_(TS⌈ö≢) ⍝ vector product
```

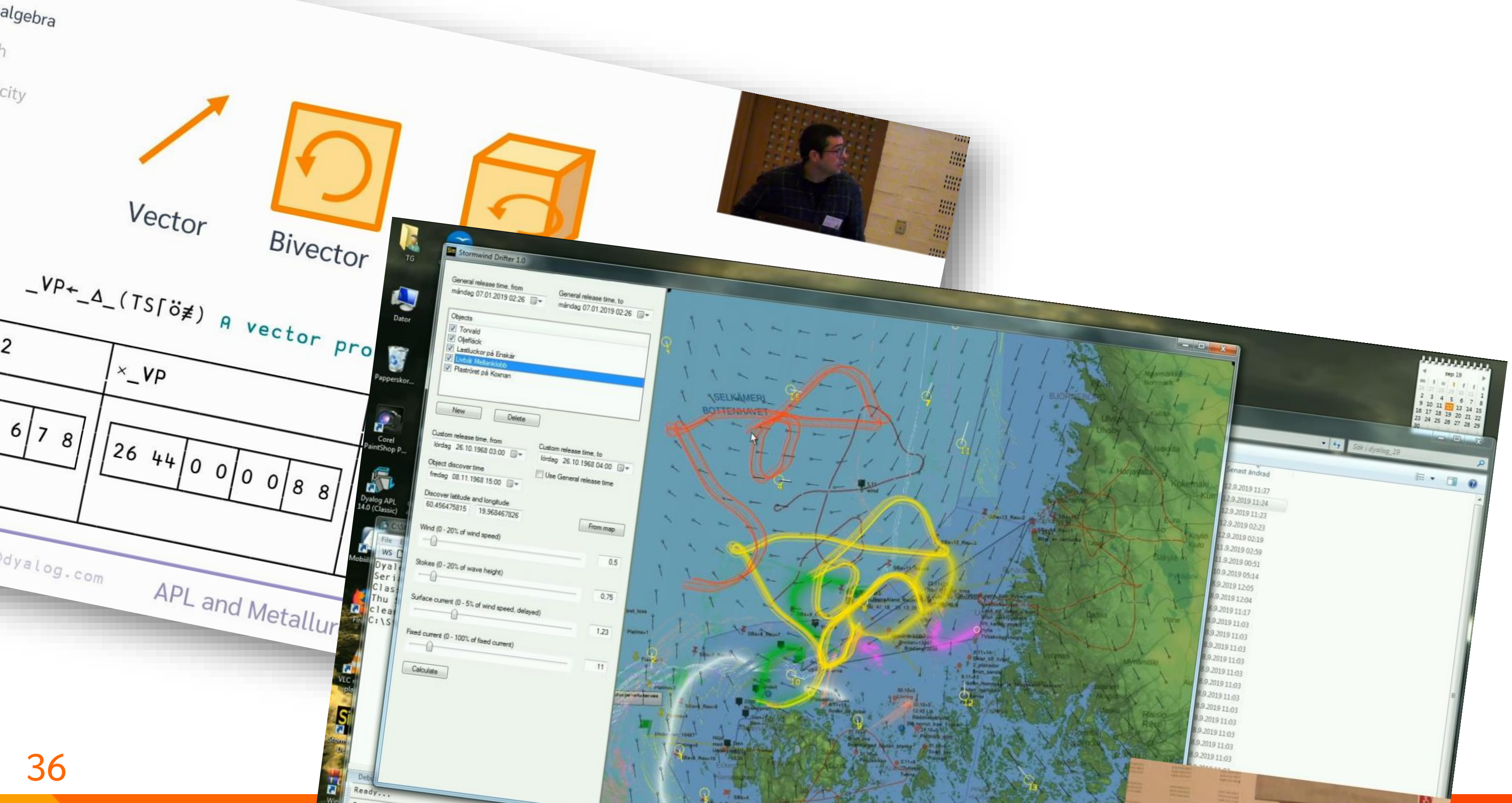
```
2          ×_VP                   ∘.×_VP                        +.×_VP
 6 7 8     26 44 0 0 0 0 8 8      26 30 0 0 0 0 8 10            70 0 0 16
                                  38 44 0 0 0 0 6  8
```

dyalog.com

APL and Metallurgy

jesus.galanlopez@ugent.be

APL in Research



algebra

city

```
_VP←_Δ_(TS⌈ö≢) ⍝ vector pro
```

2	×_VP
6 7 8	26 44 0 0 0 0 8 8

dyalog.com

APL and Metallur

Stormwind Drifter 1.0

General release time, from: måndag 07.01.2019 02:26

General release time, to: måndag 07.01.2019 02:26

Objects

- ☑ Torvald
- ☑ Oljefläck
- ☑ Lastluckor på Enskär
- ☑ Livbåt Mellanklobb
- ☑ Plaströret på Koxnan

Custom release time, from: lördag 26.10.1968 03:00

Custom release time, to: lördag 26.10.1968 04:00

Object discover time: fredag 08.11.1968 15:00

□ Use General release time

Discover latitude and longitude: 60,456475815 19,968467826

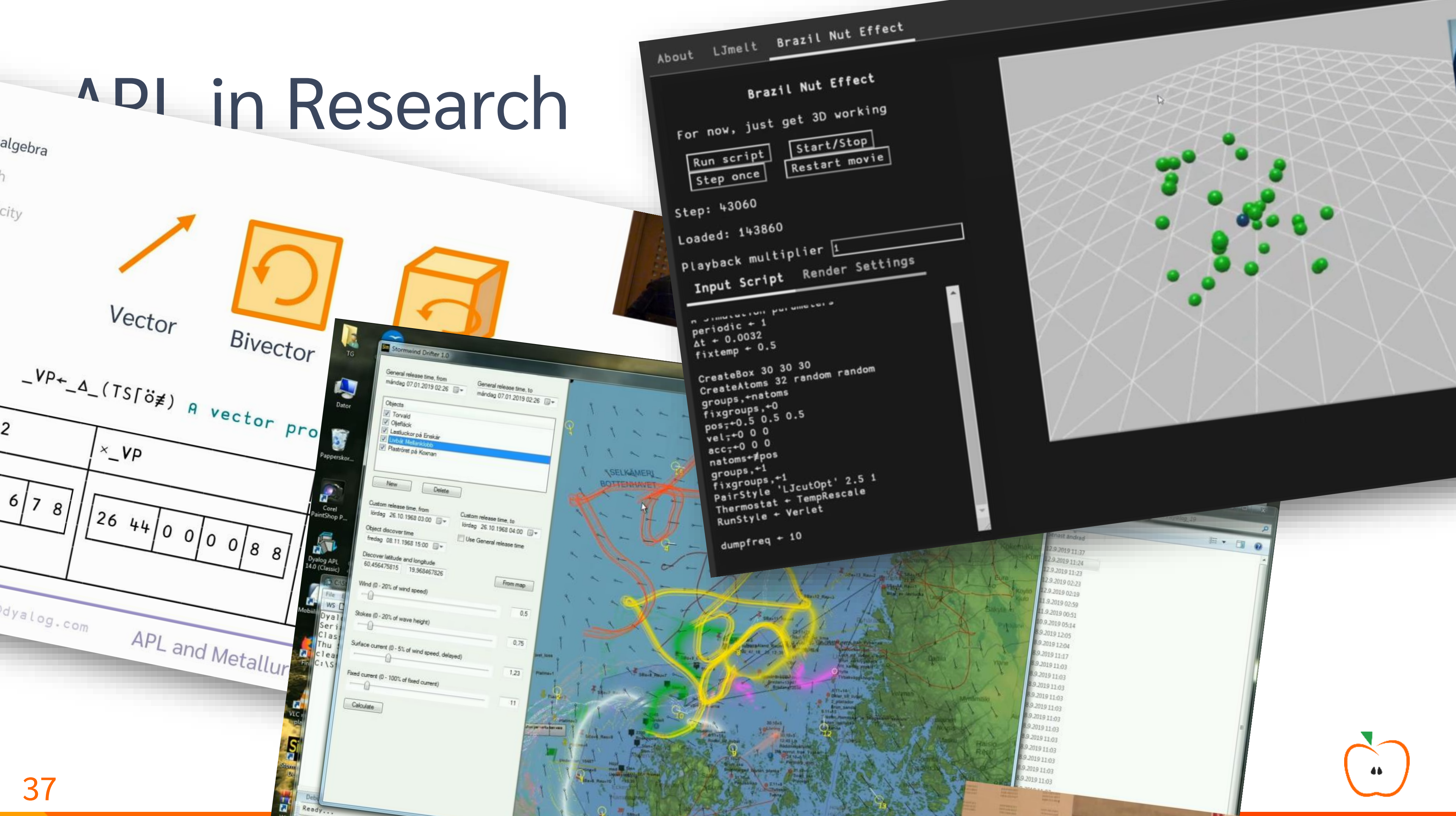
Wind (0 - 20% of wind speed): 0,5

Stokes (0 - 20% of wave height): 0,75

Surface current (0 - 5% of wind speed, delayed): 1,23

Fixed current (0 - 100% of fixed current): 11

APL in Research



```
_VP←_∆_(TS⌈ö≢) ⍝ vector pro
```

2	×_VP
6 7 8	26 44 0 0 0 0 8 8

Stormwind Drifter 1.0

General release time, from: måndag 07.01.2019 02:26

General release time, to: måndag 07.01.2019 02:26

Objects
- ☑ Torvald
- ☑ Oljefläck
- ☑ Lastluckor på Enskär
- ☑ Livbåt Mellankobb
- ☑ Plaströret på Koxnan

Custom release time, from: lördag 26.10.1968 03:00

Custom release time, to: lördag 26.10.1968 04:00

□ Use General release time

Object discover time: fredag 08.11.1968 15:00

Discover latitude and longitude: 60,456475815 19,968467826

Wind (0 - 20% of wind speed): 0,5

Stokes (0 - 20% of wave height): 0,75

Surface current (0 - 5% of wind speed, delayed): 1,23

Fixed current (0 - 100% of fixed current): 11

Brazil Nut Effect

For now, just get 3D working

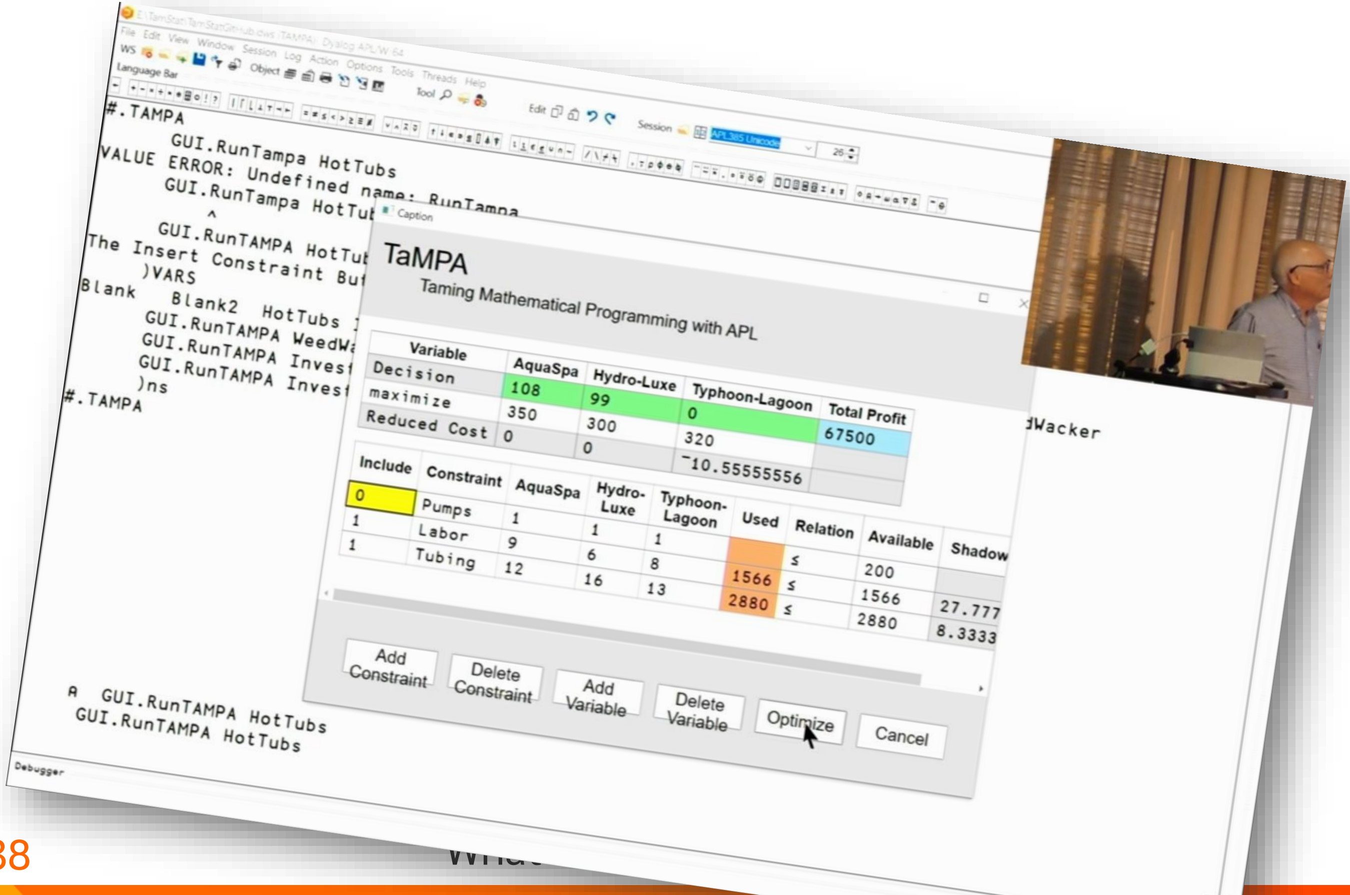
Step: 43060

Loaded: 143860

Playback multiplier: 1

```
periodic ← 1
∆t ← 0.0032
fixtemp ← 0.5

CreateBox 30 30 30
CreateAtoms 32 random random
groups,←natoms
fixgroups,←0
pos⍪←0.5 0.5 0.5
vel⍪←0 0 0
acc⍪←0 0 0
natoms←≢pos
groups,←1
fixgroups,←1
PairStyle 'LJcutOpt' 2.5 1
Thermostat ← TempRescale
RunStyle ← Verlet

dumpfreq ← 10
```

APL in Academia



File Edit View Window Session Log Action Options Tools Threads Help

```
#.TAMPA
      GUI.RunTampa HotTubs
VALUE ERROR: Undefined name: RunTampa
      GUI.RunTampa HotTub
         ^
      GUI.RunTAMPA HotTub
The Insert Constraint Bu
      )VARS
Blank   Blank2  HotTubs I                    dWacker
      GUI.RunTAMPA WeedWa
      GUI.RunTAMPA Invest
      GUI.RunTAMPA Invest
      )ns
#.TAMPA


 ⍝  GUI.RunTAMPA HotTubs
  GUI.RunTAMPA HotTubs
```

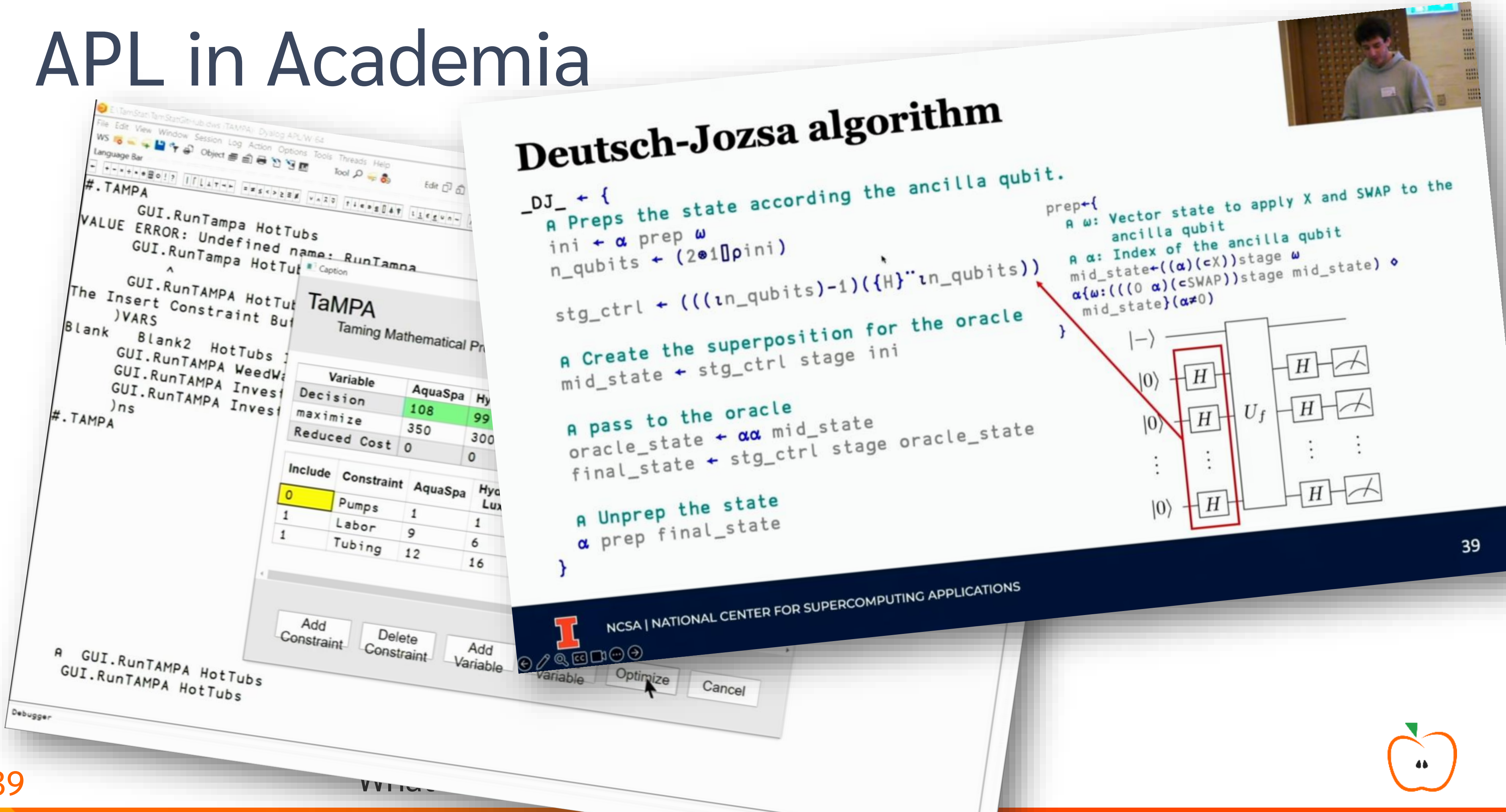
TaMPA

Taming Mathematical Programming with APL

Variable	AquaSpa	Hydro-Luxe	Typhoon-Lagoon	Total Profit
Decision	108	99	0	
maximize	350	300	320	67500
Reduced Cost	0	0	¯10.55555556	

Include	Constraint	AquaSpa	Hydro-Luxe	Typhoon-Lagoon	Used	Relation	Available	Shadow
0	Pumps	1	1	1		≤	200	
1	Labor	9	6	8	1566	≤	1566	27.777
1	Tubing	12	16	13	2880	≤	2880	8.3333

APL in Academia



File Edit View Window Session Log Action Options Tools Threads Help
WS Object Tool Edit
Language Bar
Caption
Debugger

```
#.TAMPA
      GUI.RunTampa HotTubs
VALUE ERROR: Undefined name: RunTampa
      GUI.RunTampa HotTub
      ^
      GUI.RunTAMPA HotTub
The Insert Constraint But
      )VARS
Blank   Blank2  HotTubs
      GUI.RunTAMPA WeedWa
      GUI.RunTAMPA Invest
      GUI.RunTAMPA Invest
      )ns
#.TAMPA
```

```
⍝ GUI.RunTAMPA HotTubs
GUI.RunTAMPA HotTubs
```

TaMPA

Taming Mathematical Pr

Variable	AquaSpa	Hy
Decision	108	99
maximize	350	300
Reduced Cost	0	0

Include	Constraint	AquaSpa	Hyd Lux
0	Pumps	1	1
1	Labor	9	6
1	Tubing	12	16

Deutsch-Jozsa algorithm

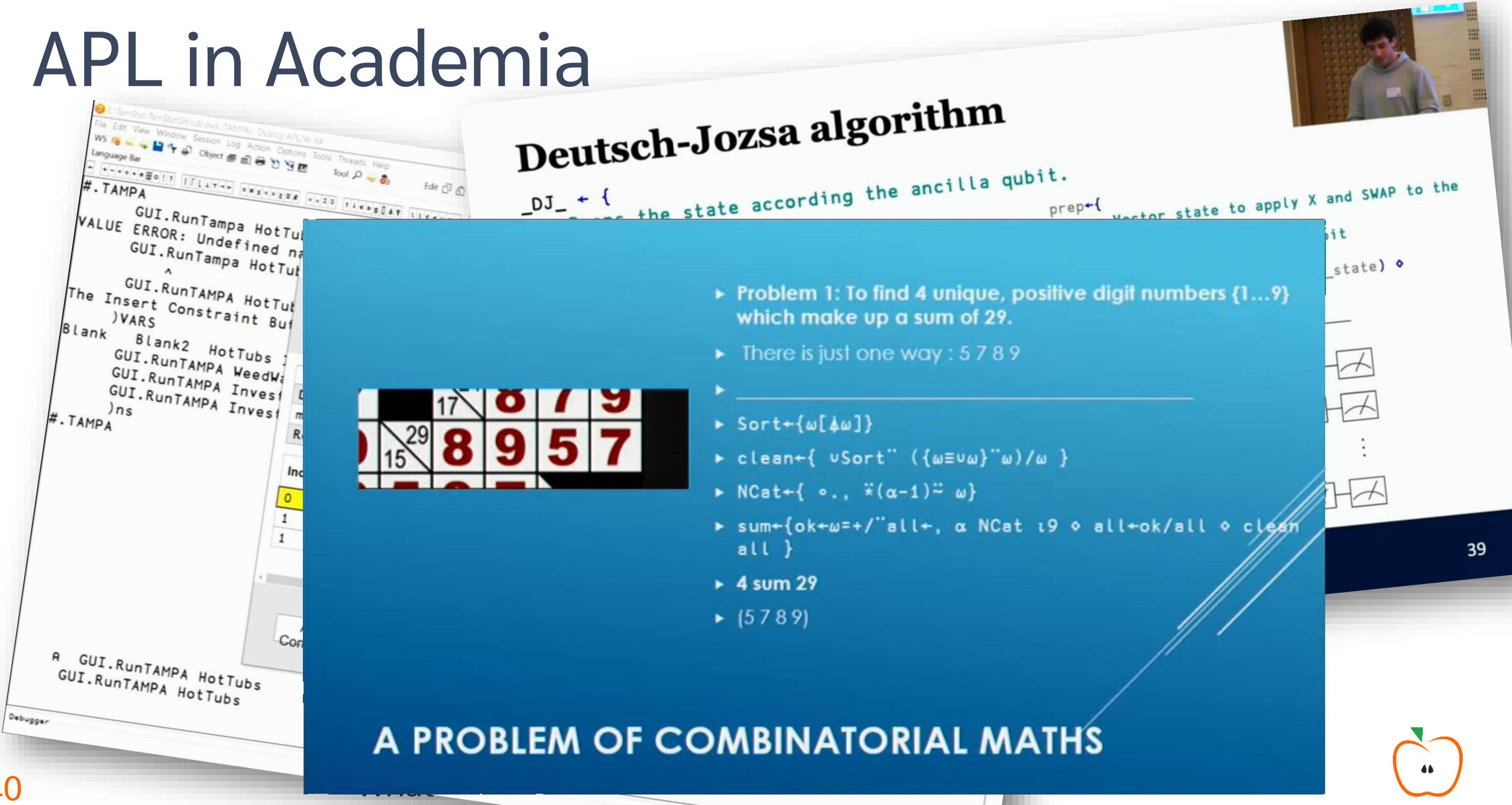
```
_DJ_ ← {
  ⍝ Preps the state according the ancilla qubit.
  ini ← ⍺ prep ⍵
  n_qubits ← (2⍟1⌷⍴ini)

  stg_ctrl ← (((⍳n_qubits)-1)({H}¨⍳n_qubits))

  ⍝ Create the superposition for the oracle
  mid_state ← stg_ctrl stage ini

  ⍝ pass to the oracle
  oracle_state ← ⍺⍺ mid_state
  final_state ← stg_ctrl stage oracle_state

  ⍝ Unprep the state
  ⍺ prep final_state
}
```

```
prep←{
  ⍝ ⍵: Vector state to apply X and SWAP to the
       ancilla qubit
  ⍝ ⍺: Index of the ancilla qubit
  mid_state←((⍺)(⊂X))stage ⍵
  ⍺{⍵:(((0 ⍺)(⊂SWAP))stage mid_state) ⋄
   mid_state}(⍺≠0)
}
```

NCSA | NATIONAL CENTER FOR SUPERCOMPUTING APPLICATIONS

39

APL in Academia



```
#.TAMPA
      GUI.RunTampa HotTu
VALUE ERROR: Undefined na
      GUI.RunTampa HotTu
         ^
      GUI.RunTAMPA HotTu
The Insert Constraint Bu
      )VARS
Blank   Blank2  HotTubs
      GUI.RunTAMPA WeedW
      GUI.RunTAMPA Inves
      GUI.RunTAMPA Inves
      )ns
#.TAMPA
```

```
A  GUI.RunTAMPA HotTubs
   GUI.RunTAMPA HotTubs
```

Deutsch-Jozsa algorithm

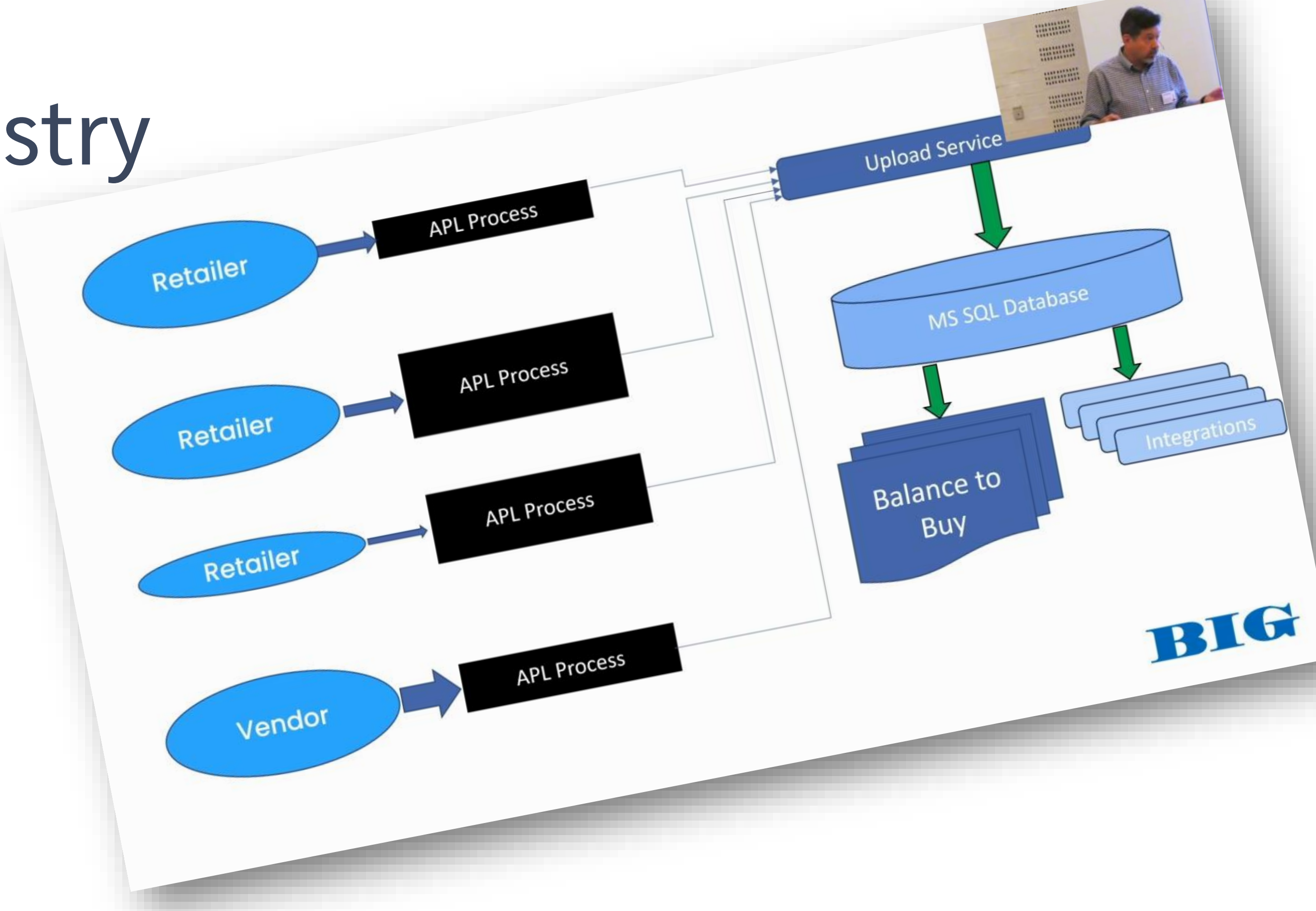
```
_DJ_ ← {
   the state according the ancilla qubit.
prep←{  Vector state to apply X and SWAP to the
_state) ⋄
```

39

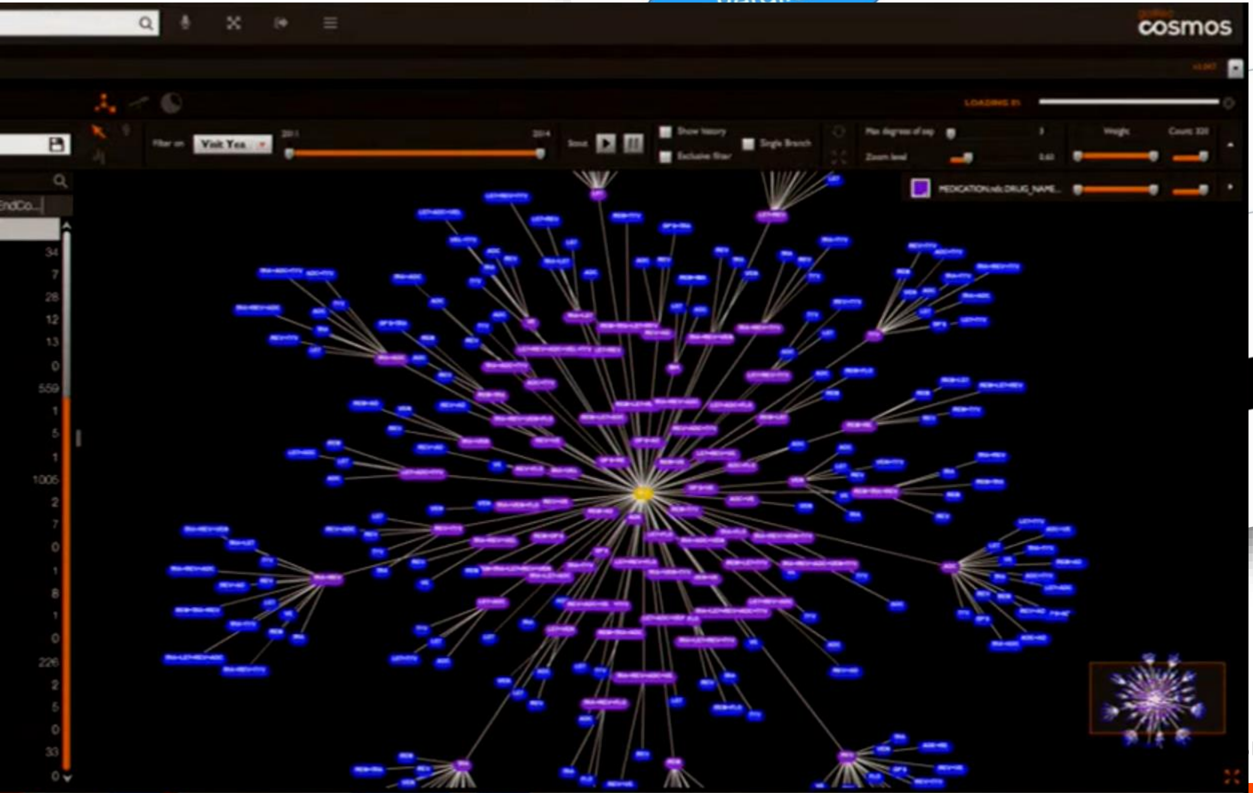
- Problem 1: To find 4 unique, positive digit numbers {1…9} which make up a sum of 29.
- There is just one way : 5 7 8 9
- ____
- Sort←{⍵[⍋⍵]}
- clean←{ ∪Sort¨ ({⍵≡∪⍵}¨⍵)/⍵ }
- NCat←{ ∘., ⍣(⍺-1)⍨ ⍵}
- sum←{ok←⍵=+/¨all←, ⍺ NCat ⍳9 ⋄ all←ok/all ⋄ clean all }
- 4 sum 29
- (5 7 8 9)

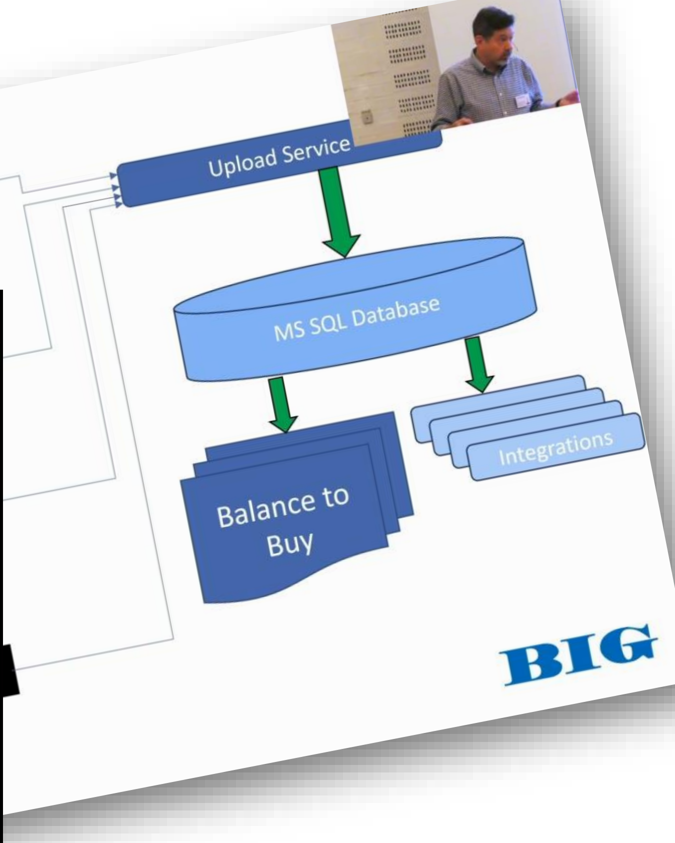
A PROBLEM OF COMBINATORIAL MATHS

APL in Industry



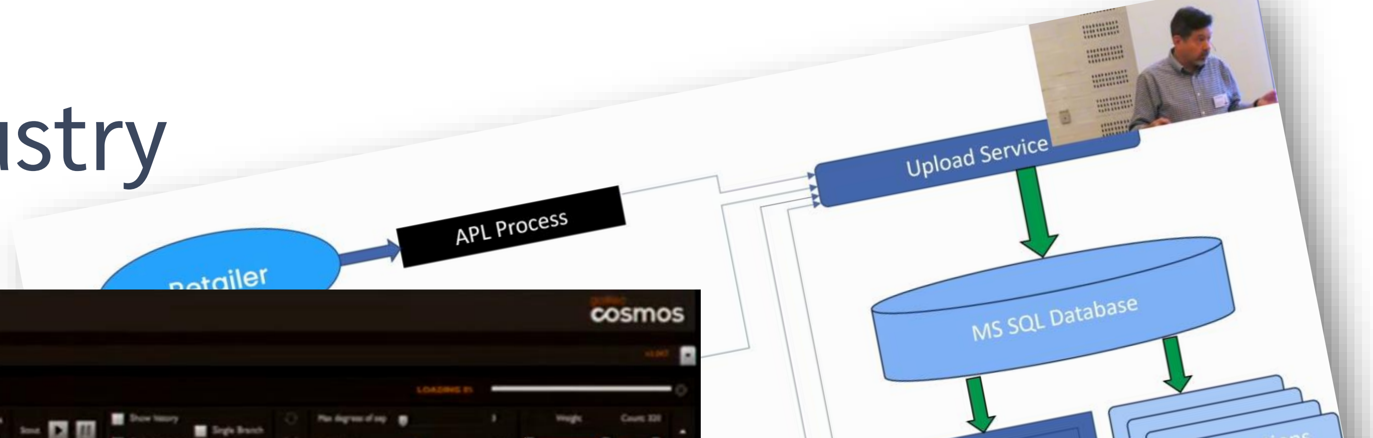
APL in Industry

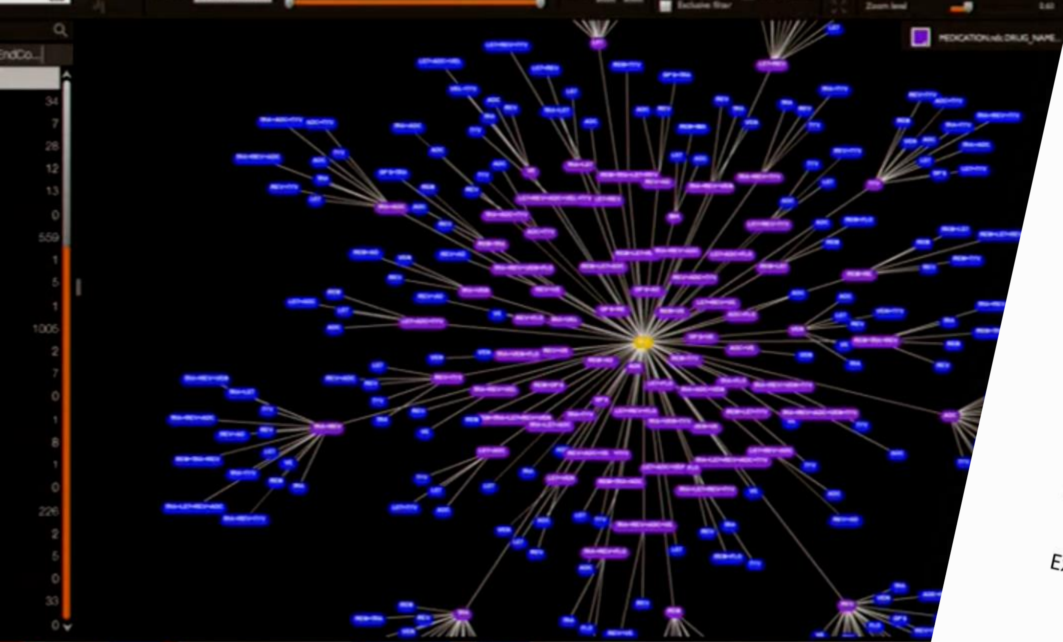


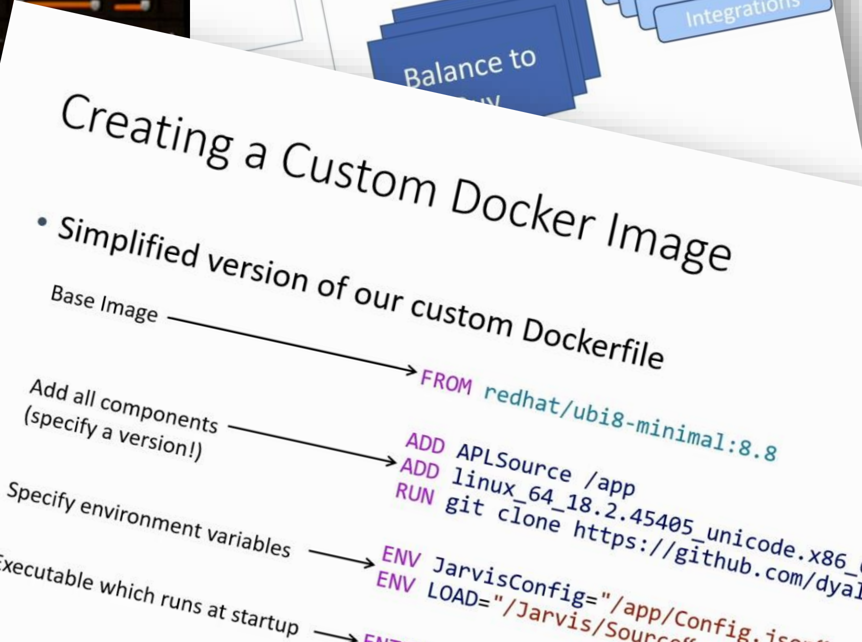


u?

APL in Industry







Creating a Custom Docker Image

- Simplified version of our custom Dockerfile

Base Image →

```
FROM redhat/ubi8-minimal:8.8
```

Add all components (specify a version!) →

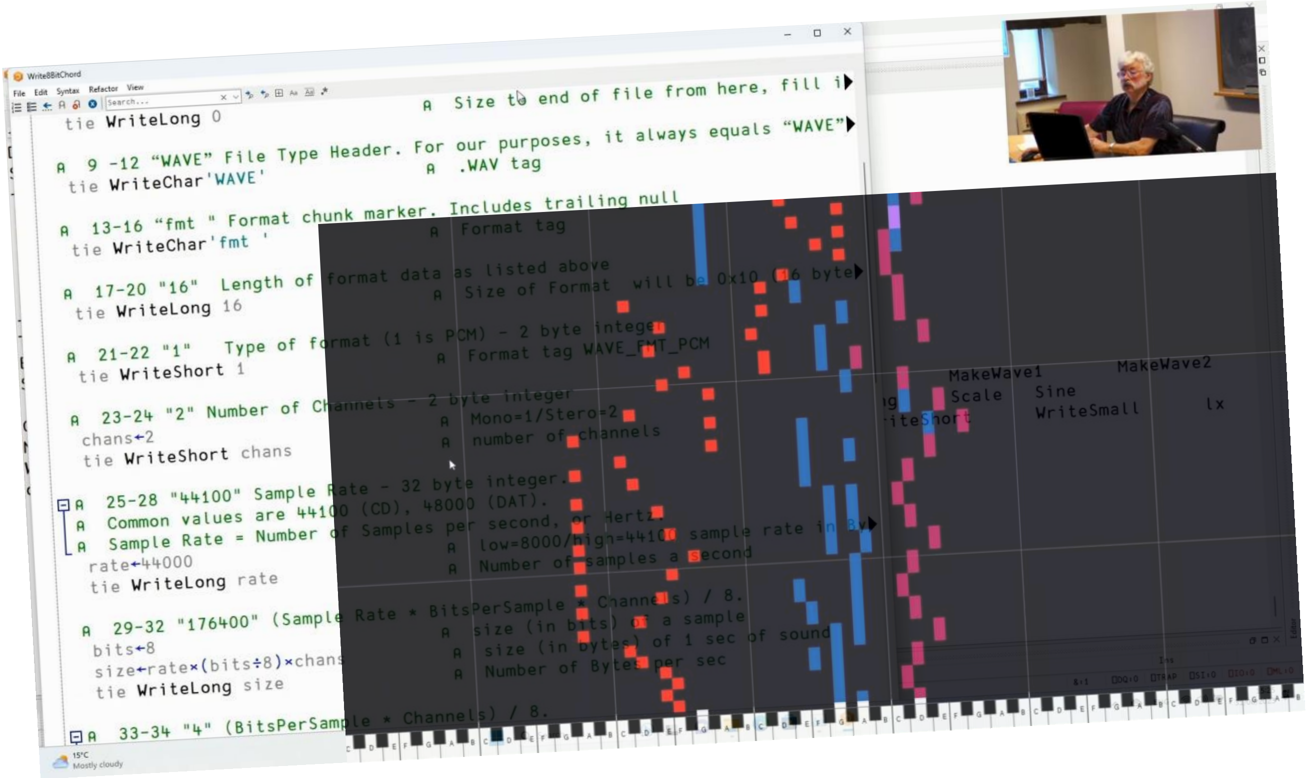
```
ADD APLSource /app
ADD linux_64_18.2.45405_unicode.x86_
RUN git clone https://github.com/dyal
```

Specify environment variables →

```
ENV JarvisConfig="/app/Config.json
ENV LOAD="/Jarvis/Sourc
```

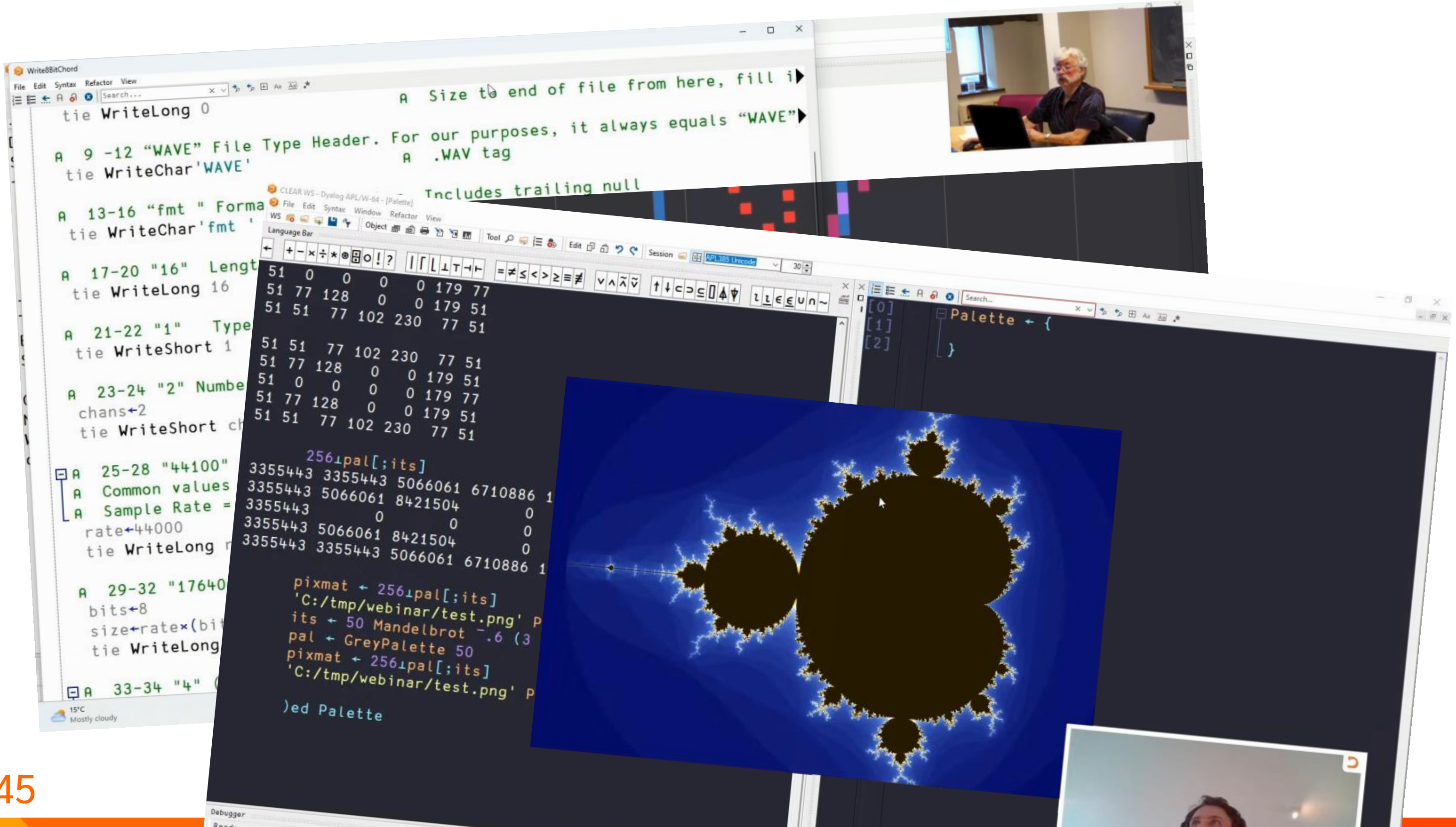
Executable which runs at startup →

And more…



```
 tie WriteLong 0                               ⍝  Size to end of file from here, fill i
⍝  9 -12 “WAVE” File Type Header. For our purposes, it always equals “WAVE”
 tie WriteChar'WAVE'                           ⍝  .WAV tag

⍝  13-16 “fmt " Format chunk marker. Includes trailing null
 tie WriteChar'fmt '                           ⍝  Format tag

⍝  17-20 "16"  Length of format data as listed above
 tie WriteLong 16                              ⍝  Size of Format  will be 0x10 (16 byte

⍝  21-22 "1"   Type of format (1 is PCM) - 2 byte integer
 tie WriteShort 1                              ⍝  Format tag WAVE_FMT_PCM

⍝  23-24 "2" Number of Channels - 2 byte integer
 chans←2                                       ⍝  Mono=1/Stero=2
 tie WriteShort chans                          ⍝  number of channels

⍝  25-28 "44100" Sample Rate - 32 byte integer.
⍝  Common values are 44100 (CD), 48000 (DAT).
⍝  Sample Rate = Number of Samples per second, or Hertz.
 rate←44000                                    ⍝  low=8000/high=44100 sample rate in By
 tie WriteLong rate                            ⍝  Number of samples a second

⍝  29-32 "176400" (Sample Rate * BitsPerSample * Channels) / 8.
 bits←8                                        ⍝  size (in bits) of a sample
 size←rate×(bits÷8)×chans                      ⍝  size (in bytes) of 1 sec of sound
 tie WriteLong size                            ⍝  Number of Bytes per sec

⍝  33-34 "4" (BitsPerSample * Channels) / 8.
```

And more...



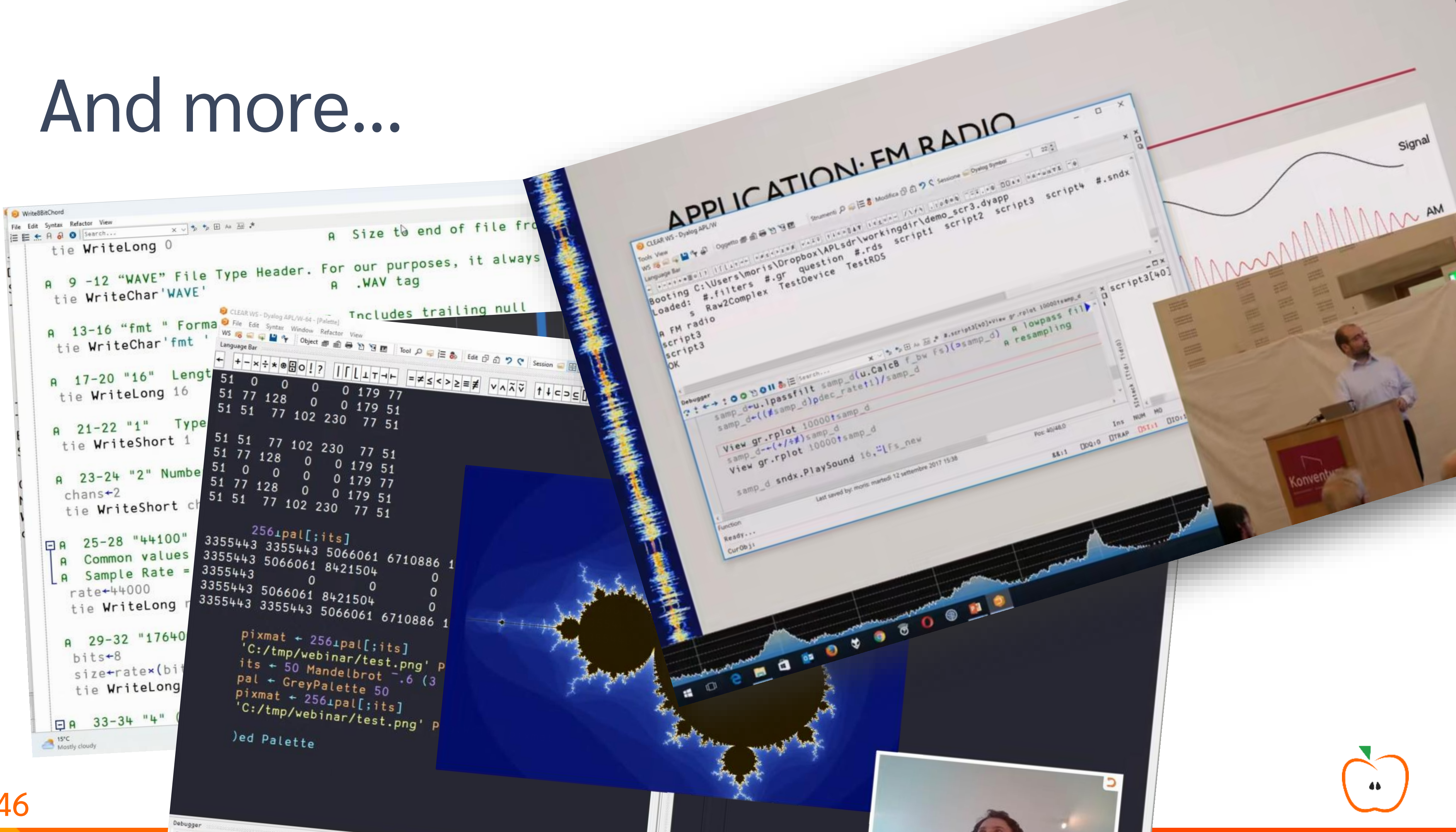
```
tie WriteLong 0                          ⍝  Size to end of file from here, fill i

⍝  9 -12 “WAVE” File Type Header. For our purposes, it always equals “WAVE”
 tie WriteChar'WAVE'                     ⍝  .WAV tag

                                            Includes trailing null
⍝  13-16 “fmt " Forma
 tie WriteChar'fmt '

⍝  17-20 "16"  Lengt
 tie WriteLong 16

⍝  21-22 "1"   Type
 tie WriteShort 1

⍝  23-24 "2" Numbe
 chans←2
 tie WriteShort ch

⍝  25-28 "44100"
⍝  Common values
⍝  Sample Rate =
 rate←44000
 tie WriteLong r

⍝  29-32 "17640
 bits←8
 size←rate×(bi
 tie WriteLong

⍝  33-34 "4" (
```

```
51  0   0   0   0 179 77
51 77 128   0   0 179 51
51 51  77 102 230  77 51

51 51  77 102 230  77 51
51 77 128   0   0 179 51
51  0   0   0   0 179 77
51 77 128   0   0 179 51
51 51  77 102 230  77 51

      256⊥pal[;its]
3355443 3355443 5066061 6710886 1
3355443 5066061 8421504       0
3355443       0       0       0
3355443 5066061 8421504       0
3355443 3355443 5066061 6710886 1

      pixmat ← 256⊥pal[;its]
      'C:/tmp/webinar/test.png' p
      its ← 50 Mandelbrot ¯.6 (3
      pal ← GreyPalette 50
      pixmat ← 256⊥pal[;its]
      'C:/tmp/webinar/test.png' p

      )ed Palette
```

```
[0]   Palette ← {
[1]
[2]   }
```

Write8BitChord
File Edit Syntax Refactor View
Search...
CLEAR WS - Dyalog APL/W-64 - [Palette]
File Edit Syntax Window Refactor View
WS Object Tool Edit Session
Language Bar
APL385 Unicode
30
15°C
Mostly cloudy
Debugger

And more...



Write8BitChord

```
tie WriteLong 0                              ⍝  Size to end of file fro

⍝  9 -12 “WAVE” File Type Header. For our purposes, it always
 tie WriteChar'WAVE'                         ⍝  .WAV tag

                                                Includes trailing null
⍝  13-16 “fmt " Forma
 tie WriteChar'fmt '

⍝  17-20 "16"  Lengt
 tie WriteLong 16

⍝  21-22 "1"   Type
 tie WriteShort 1

⍝  23-24 "2" Numbe
 chans←2
 tie WriteShort ch

⍝  25-28 "44100"
⍝  Common values
⍝  Sample Rate =
 rate←44000
 tie WriteLong r

⍝  29-32 "17640
 bits←8
 size←rate×(bit
 tie WriteLong

⍝  33-34 "4" (
```

CLEAR WS - Dyalog APL/W-64 - [Palette]

```
51  0   0   0   0 179 77
51 77 128   0   0 179 51
51 51  77 102 230  77 51

51 51  77 102 230  77 51
51 77 128   0   0 179 51
51  0   0   0   0 179 77
51 77 128   0   0 179 51
51 51  77 102 230  77 51

      256⊥pal[;its]
3355443 3355443 5066061 6710886 1
3355443 5066061 8421504       0
3355443       0       0       0
3355443 5066061 8421504       0
3355443 3355443 5066061 6710886 1

      pixmat ← 256⊥pal[;its]
      'C:/tmp/webinar/test.png' P
      its ← 50 Mandelbrot ¯.6 (3
      pal ← GreyPalette 50
      pixmat ← 256⊥pal[;its]
      'C:/tmp/webinar/test.png' P

      )ed Palette
```

APPLICATION: FM RADIO

CLEAR WS - Dyalog APL/W

```
Booting C:\Users\moris\Dropbox\APLsdr\workingdir\demo_scr3.dyapp
Loaded:  #.filters  #.gr  question  #.rds  script1  script2  script3  script4  #.sndx
      s  Raw2Complex  TestDevice  TestRDS
⍝ FM radio
script3
script3
OK
```

```
samp_d←u.lpassfilt samp_d(u.CalcB f_bw fs)(⊃samp_d)  ⍝ lowpass fil
samp_d←((≢samp_d)⍴dec_rate↑1)/samp_d                 ⍝ resampling
View gr.rplot 10000↑samp_d
samp_d-←(+/÷≢)samp_d
View gr.rplot 10000↑samp_d
samp_d sndx.PlaySound 16,⌊fs_new
```

Last saved by: moris martedì 12 settembre 2017 15:38

Getting Started and Learning APL

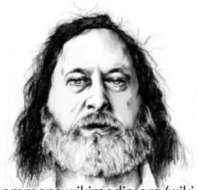
Rho, rho, rho of X

Always equals 1

Rho is dimension, rho rho rank

APL is fun!

— *Richard Stallman*



commons.wikimedia.org/wiki/
File:Retrat_Richard_Stallman.jpg

```
      ⍴⍴⍴X
1
      ⍴X
3 2 4
      ⍴⍴X
3
      ⍴⍴⍴X
1
```

Getting Started and Learning APL

Rho, rho, rh

ρX

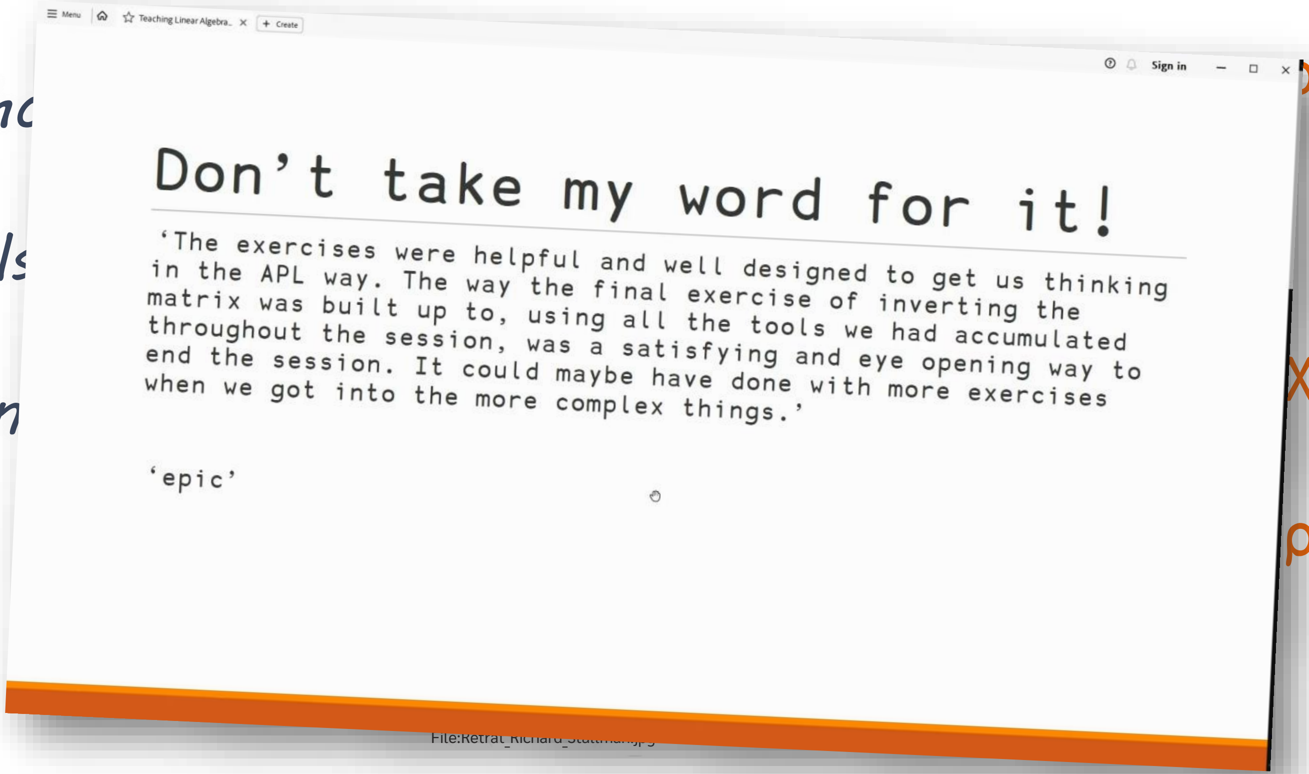
Always equals

X

Rho is dimen

ρX

APL is fun!

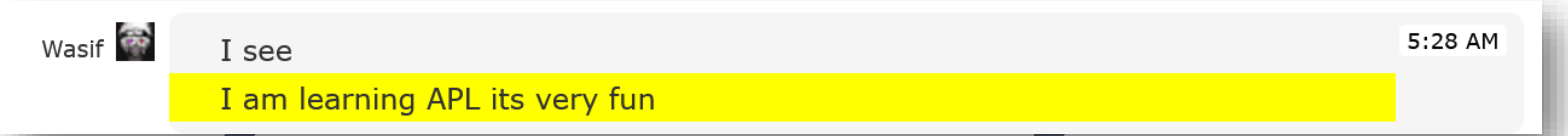


Don't take my word for it!

'The exercises were helpful and well designed to get us thinking in the APL way. The way the final exercise of inverting the matrix was built up to, using all the tools we had accumulated throughout the session, was a satisfying and eye opening way to end the session. It could maybe have done with more exercises when we got into the more complex things.'

'epic'

File:Retrat_Richard_Stallman.jpg

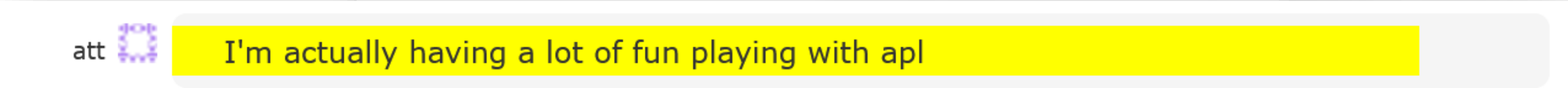


Wasif

I see

I am learning APL its very fun

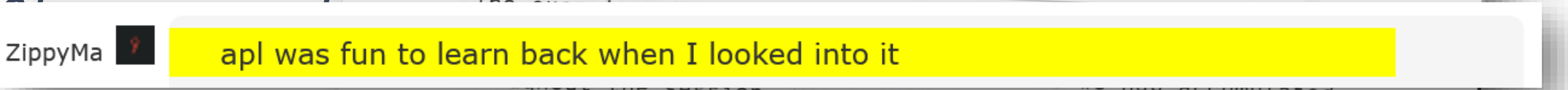
5:28 AM



att

I'm actually having a lot of fun playing with apl

Don't take my word for it!

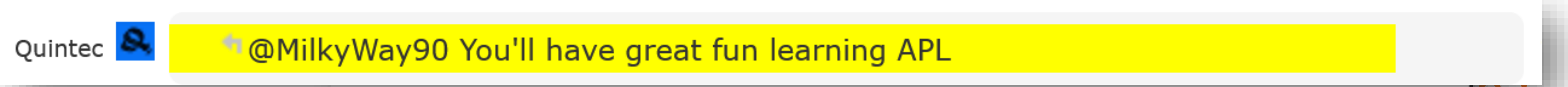


ZippyMa

apl was fun to learn back when I looked into it

end the session. It could maybe have done with more exercises when we got into the more complex things.'

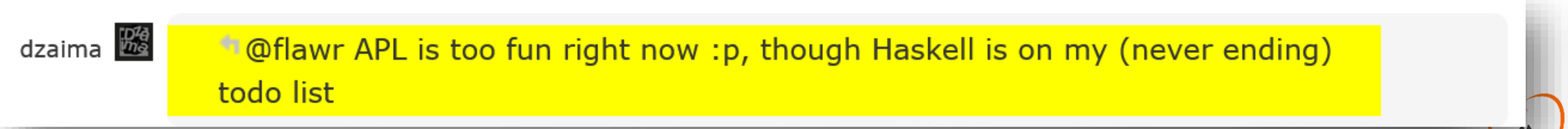
Rho is dimen



Quintec

@MilkyWay90 You'll have great fun learning APL

APL is fun!



dzaima

@flawr APL is too fun right now :p, though Haskell is on my (never ending) todo list