Reinvigorating STARMAP

Who am I?

Neil Kirsopp – just another JS hacker

…sometimes likes to write some APL

Who am I?
(according to the internet)

Never fully trust the machines

[I am not Martina]

What is STARMAP?

- It’s a wonderful little book
- Originally written in 1973
  - Collaboration with an astronomer
  - IBM technical demo
- The text I’ve worked from is from 1978
  - Published as a book/pamphlet
- It uses a plotter!
  - We need a new UI

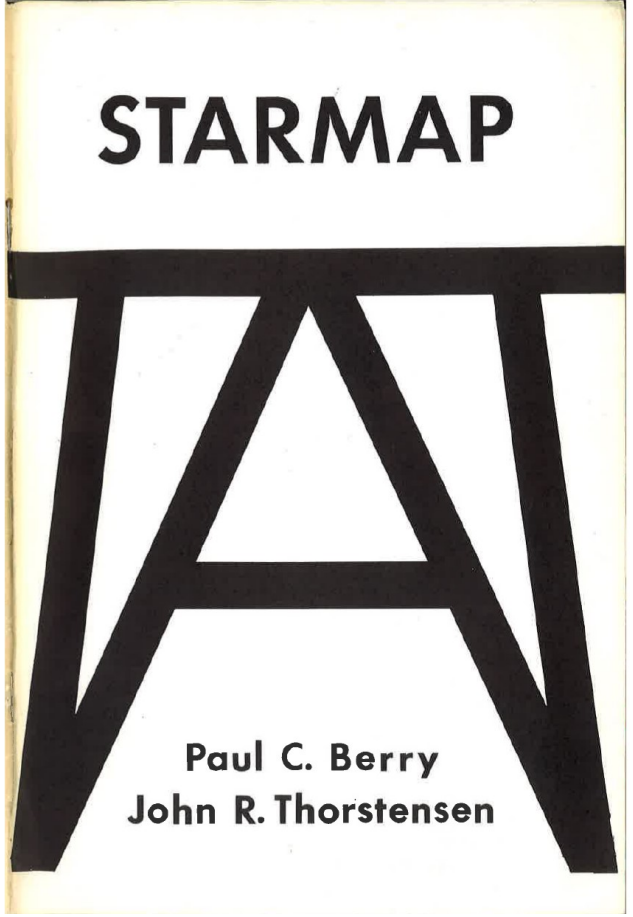
What is STARMAP, the repository?

- Transcribed by Stephen Taylor using his physical copy
  - He used Claude
  - There may have been some transcription issues
    - I used Claude to fix those

What is STARMAP?

- The teaching method is wonderful
  - I'd recommend anyone to read it
  - It teaches both astronomy and APL
    - I think this is a sweet spot for APL
- But the source originates from IBM APL
  - Work required for adapting to Dyalog APL to get calculations working
    - Before creating any visualisation

The STARMAP book



- Descriptions of the calculations
- The relevant APL
- The result of plotting

Calculating the Positions of the Planetary Bodies

The function *CALCULATEPLANETS* finds *PLANETS*, a table of the positions of the sun, moon, and planets at the desired date. When first calculated by the function *PLANETPOS*, these positions are stated in 3-dimensional heliocentric Cartesian coordinates. But the function *EARTHVIEW* converts them to geocentric polar coordinates (right ascension in hours, declination in degrees, and distance in astronomical units), locating the planets with respect to the center of the Earth.

In order to plot the sky above a particular place, the function *SKYPOS* (see p. 29) is used to calculate *AA*, a table of altiude and azimuth with respect to given time and location on the Earth's surface. The function *VISIBLE* is used to select from *P* those members that are above the horizon, and saves them in the table *AAP*. Finally, the *PROJECTION* of these coordinates onto a flat surface is calculated, and translated to the Cartesian form expected by the plotting function; the function *IF* is simply a compression of the left argument by the right, along the first axis, defined by the APL symbol ⌿.

```
∇ CALCULATEPLANETS; AA; MOON; SUN; KOHOUTEK
  PLANETCOORD←AAP←PLANETS←VP←⍳0
  PLANETS←DATE EARTHVIEW DATE PLANETPOS (3≠⍳9)⌿PLANETS
  SUN←DATE EARTHVIEW 0 0 0
  K←100≥|DATE-JNU 12 28 1973
  KOHOUTEK←DATE EARTHVIEW (DATE IF K) COMETPOS KOHOUTEK
  MOON←MOONPOS DATE
  PHASE←MOON[1;1] MOONPHASE SUN[1;1]
  PLANETS←MOON,[1] SUN,[1] PLANETS,[1] KOHOUTEK
  MOON←MOON[;3] PARALLAXADJUST (LAT,DATE,TIME) SKYPOS MOON
  AA←MOON,[1] (LAT,DATE,TIME) SKYPOS 1 0⌿PLANETS
  PLANETCOORD←MAPCARTESIAN PROJECTION AAP←AA IF VP←VISIBLE AA
```

Execution of *CALCULATEPLANETS* causes new values to be assigned to four global variables. (These are initially set to ⍳0 in the first statement, mainly to draw attention to a list of the gobal variables which will be reset as a consequence of executing this function.) The four are:

PLANETS — The right ascension and declination of the moon, sun, planets, and Kohoutek.

VP — A logical vector indicating which planets are visible from the place, date, and time requested.

AAP — The altitude and azimuth of the visible planets.

PLANETCOORD — The Cartesian coordinates used to plot the projection of the visible planets.

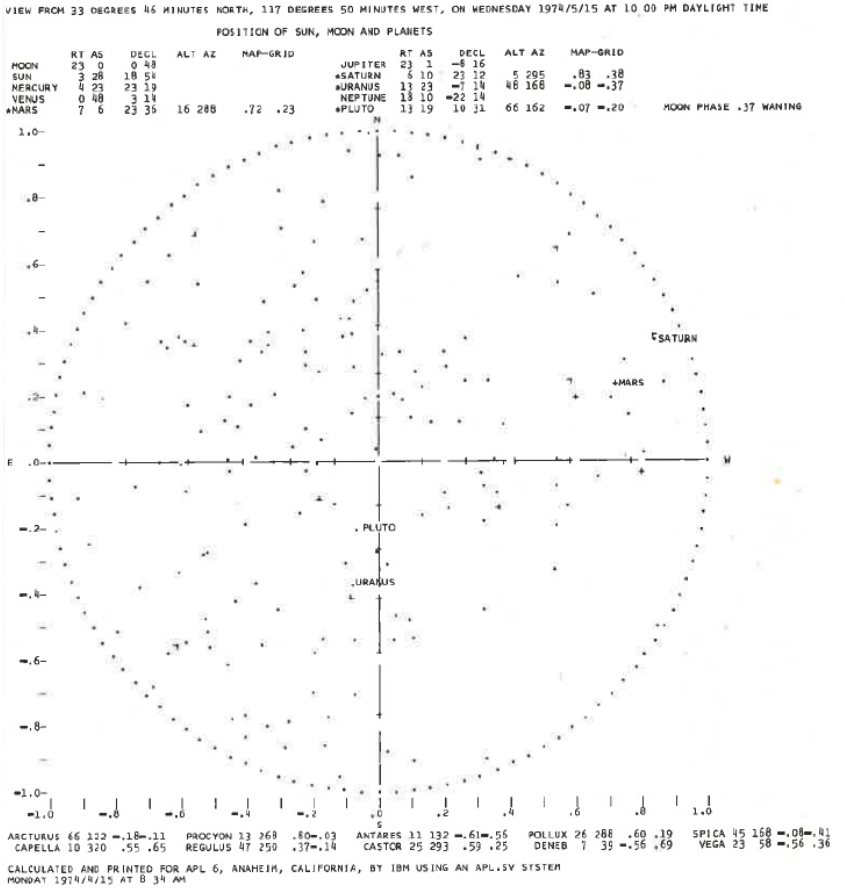


Fig. 2 Sample chart produced by the STARMAP workspace

What did I do first?

- Fixed any errors that seemed to be transcription-based

What did I do first?

- Fixed any errors that seemed to be transcription-based
- Used Claude to write tests against *current* star charts
  - Seems we were slightly disagreeing with 1973
  - Generated tests based on current JPL datasets
    - Accepted the JPL data as correct, even when it disagreed with the source

Working model, so now what?

A pre-recorded demo of a new UI that sits on top of STARMAP follows:

- Input via a ‘friendly’ browser form
- Now animated over a period of time
- Visual confirmation that it seems to work
  - Celestial bodies seem to move at likely relative speed – note the moon
- Available at https://starmap.kirsopp.me

STARMAP animation generator

Start date

15/05/1974

End date

12/06/1975

UTC time (hours, decimal)

5

Frames per second

15

e.g. 5 = 05:00 UTC = 10 PM PDT

Latitude

52.526

Longitude (east +, west −)

+0.388

Large display

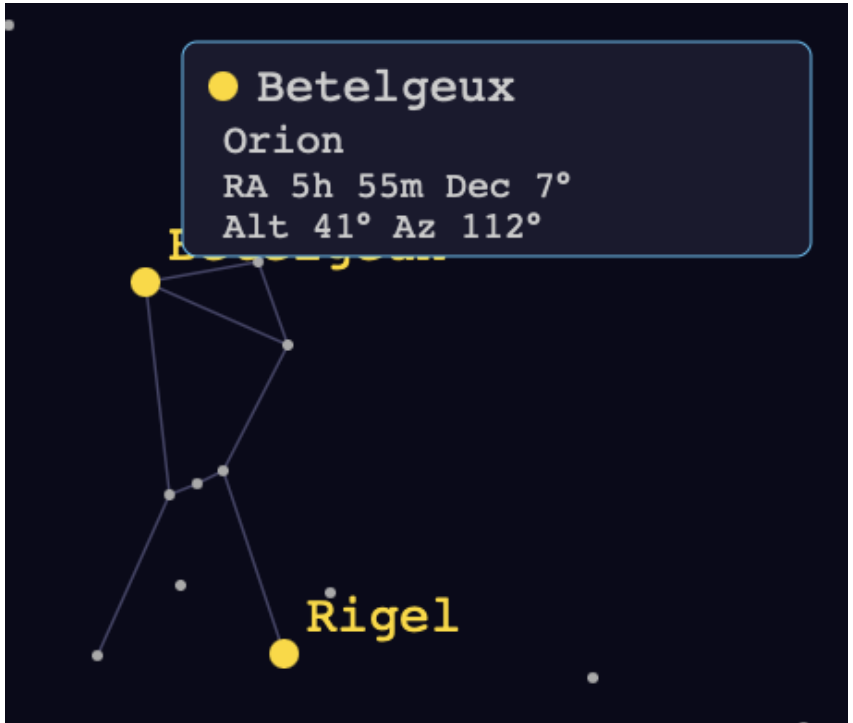
Reduce size (every other day)

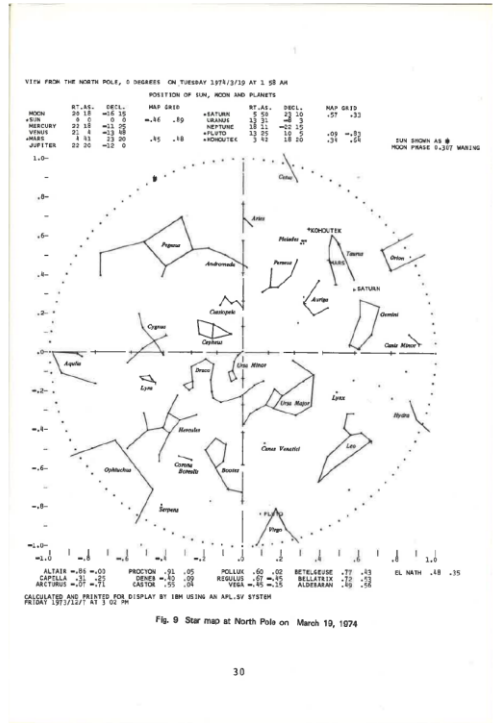
Generate SVG

Generate .mp4 (slow)

Constellations

- *”lines linking stars in the same constellation have been drawn in by hand”* – STARMAP book
  - I used freely available constellation data to automate hovering over a start to show a constellation. I’ve chosen Betelgeux, as I know that is in Orion.





Fin

Fin

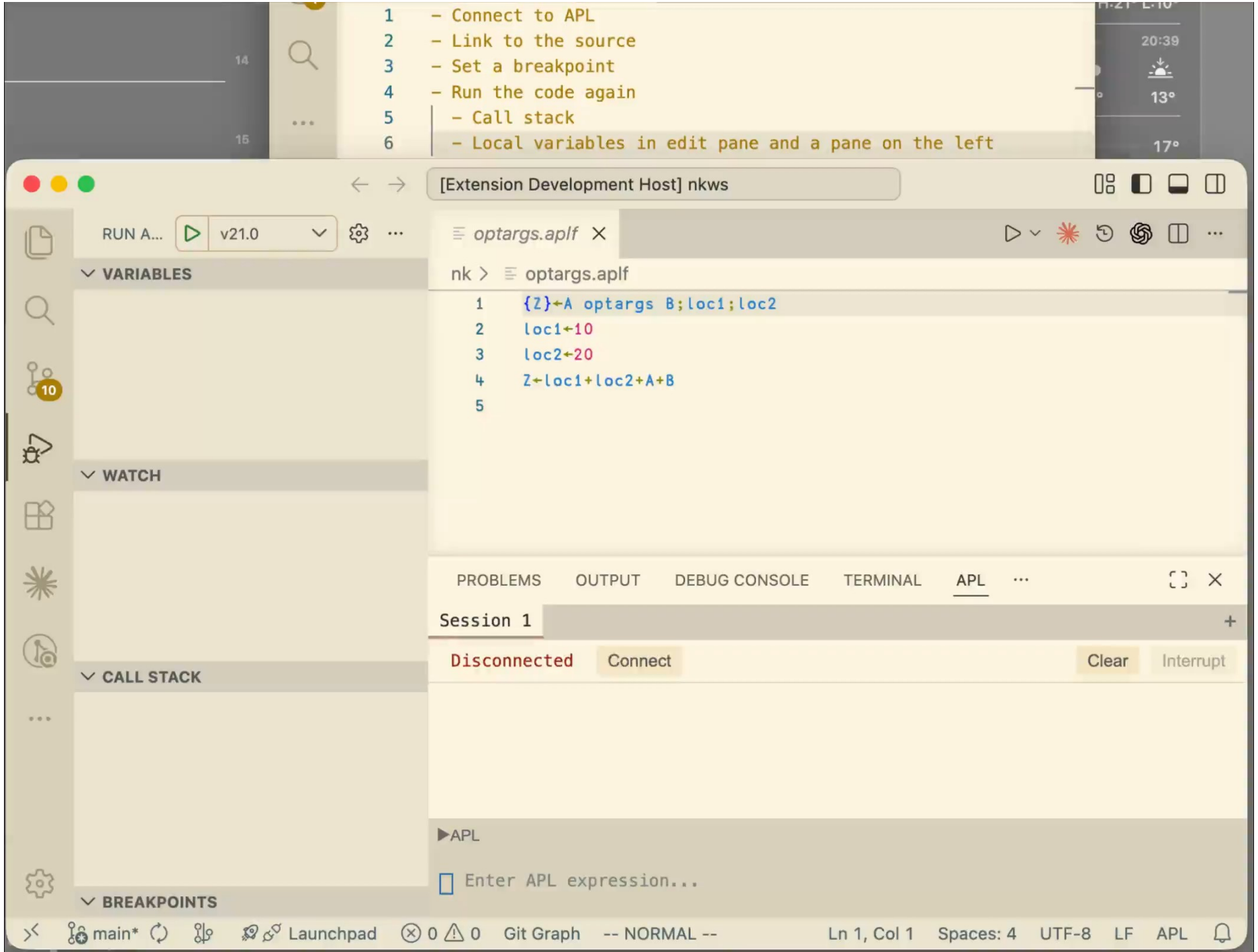
- *Claude is the new* □ - Stephen Taylor

More LLM stuff

- These slides are reserved in case I finish early…
- They are a few examples of other LLM projects I got involved in outside of work

vscode-apl

- Based on work by Gilgamesh Athoraya, developed further using LLMs by Martina Crippa
- APL debugging from within VS Code
- We are seeing more people using VS Code for their APL work
  - It is becoming standard tooling for many developers
- Short demo…


1 - Connect to APL
2 - Link to the source
3 - Set a breakpoint
4 - Run the code again
5 - Call stack
6 - Local variables in edit pane and a pane on the left
[Extension Development Host] nkws
RUN A...
v21.0
VARIABLES
WATCH
CALL STACK
BREAKPOINTS
optargs.aplf
nk > optargs.aplf
1 {Z}←A optargs B;loc1;loc2
2 loc1←10
3 loc2←20
4 Z←loc1+loc2+A+B
5
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
APL
Session 1
Disconnected
Connect
Clear
Interrupt
▶APL
Enter APL expression...
main*
Launchpad
0 0
Git Graph
-- NORMAL --
Ln 1, Col 1
Spaces: 4
UTF-8
LF
APL

gritt

- An implementation of the RIDE protocol, written in Go
- An off-shoot of a project for scripting APL
  - …but it also grew to be a terminal UI
- For LLMs, it is an easily available tool to execute APL code:

A colleague gave me a single line of APL that confused Claude

- 80+ mins to give me a full explanation through inspection
- 5 mins to give me a full explanation when enabled to execute expressions

gritt

- `gritt -l -e EXPRESSION`
  - Launches Dyalog, runs the expression and closes Dyalog
- `gritt -addr HOST:PORT -e EXPRESSION`
  - Connects to a Dyalog process and runs the expression

- Easily connecting to a persistent Dyalog instance is where the advantage lies
  - LLMs can use this to try experiments and iterate without running full programs