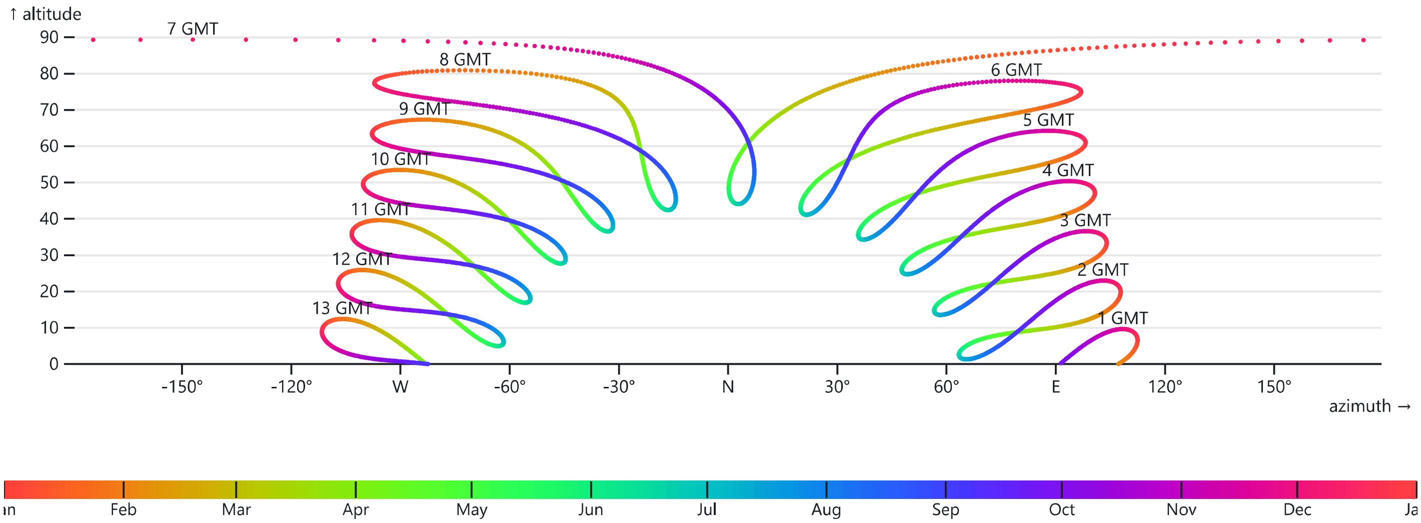
The Sun’s analemma

If you were to look at the sky every day of the year at the exact same hour, where would the Sun be? The diagram below shows its position for each hour of the day, limited by the horizon. Each twisted loop (called an analemma), is in fact made of 365 dots, one for each day of the year. The analemma shows by how much solar time is fast or slow compared to clock time, at different periods of the year.



Click on the map to set the observer’s location (defaults to Greenwich).

Longitude: 72.82 Latitude: -22.44

period

day ⌄

This diagram is facing North if the observer is in the southern hemisphere, and vice-versa. We’re using x = azimuth, y = altitude, the equirectangular projection[1].

Note how the diagram changes when you are inside the polar circle (latitude ±66°30’), or in-between the tropics (±23°26’). The loop after a year is almost perfectly closed—but you could observe a bit of drift if you extended the process over several centuries.

Looking more closely, you can see that the twisted loop the Sun does over a year is not symmetric: the January region (blue) is wider than the July region (yellow). All of this is described by the Equation of time, and coded by Vladimir Agafonkin in this tiny function that follows Jean Meeus's reference.

[1] Using the equirectangular projection makes our diagram simpler to read, but far from being a simulation of an actual picture of the sky. This would more likely require the stereographic projection—or an azimuthal equal-area projection for a fisheye camera. Something to explore in another notebook. See Mike's Solar analemmas for an example.

code

```
latitude = -22.44
```

```
center = 0
```

```
periods = ▸ Map(3) {"day" => 1, "week" => 7, "month" => 30}
```

```
  latitude = location[1]
```

```
  center = latitude < 0 ? 0 : 180
```

```
days = ▸ Array(365) [2000-01-01, 2000-01-02, 2000-01-03, 2000-01-04, 2000-01-05, 2000
```

```
  periods = new Map([
    ["day", 1],
    ["week", 7],
    ["month", 30]
  ])
```

```
  days = d3
    .utcDays(Date.UTC(2000, 0, 1), Date.UTC(2001, 0, 1), periods.get(period))
    .slice(0, -1)
```

```
hours = days.flatMap((day) => d3.utcHours(day, +day + 24 * 3600 * 1000, 1))
```

```
data = hours
  .map((date) => ({
    date,
    hour: date.getUTCHours(),
    ...getSunPosition(date, ...location)
  }))
  .filter((d) => d.altitude > 0)
```

```
import {degrees} from "@fil/math"
```

```
import { getSunPosition } from "@mourner/sun-position-in-900-bytes"
```

```
import { worldMapCoordinates } from "@enjalot/draggable-world-map-coordinates-
input"
```

```
import { degrees } from "@fil/math"
```