Common pytz and datetime gotchas

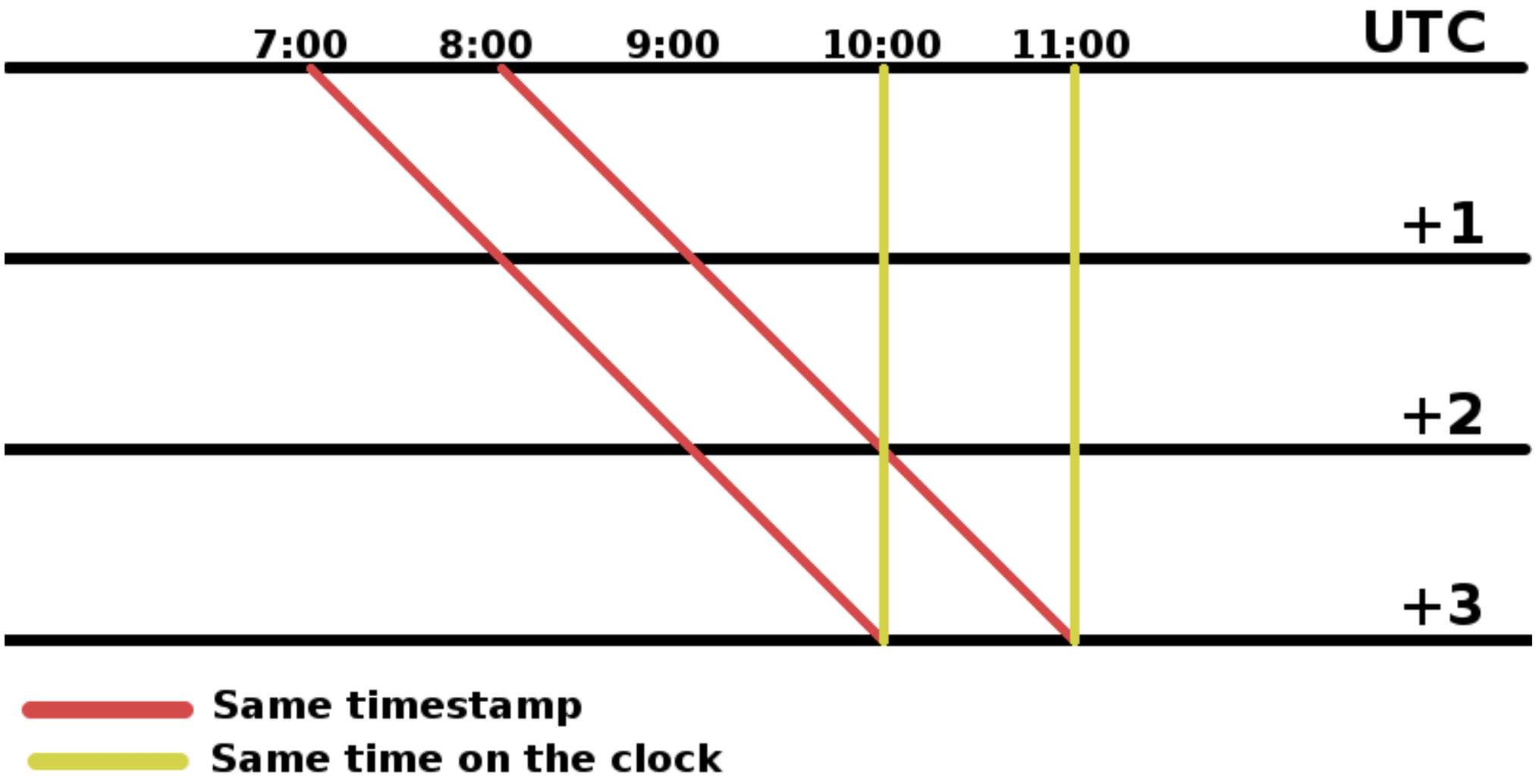
Ignas Mikalajūnas
<ignas@pov.lt>

Programmers of Vilnius
http://pov.lt/

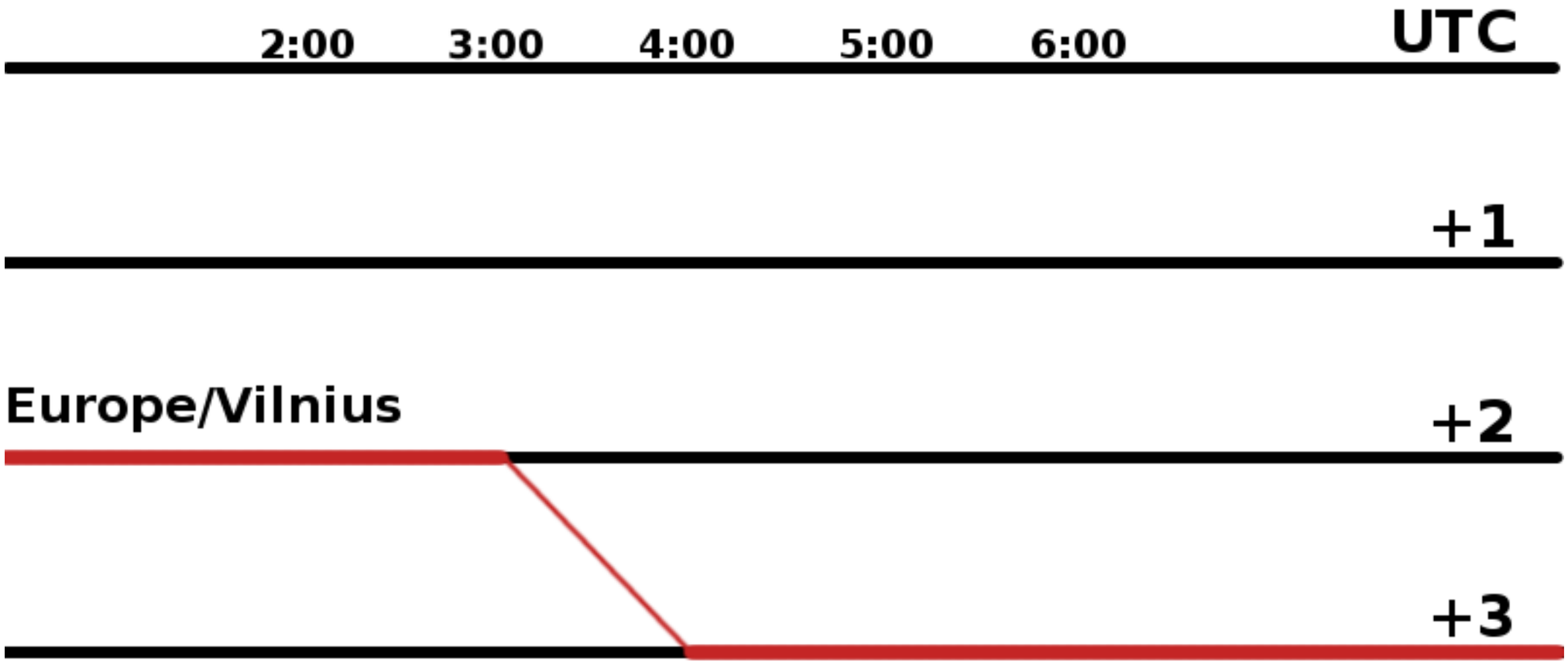
EuroPython 2006

timezones

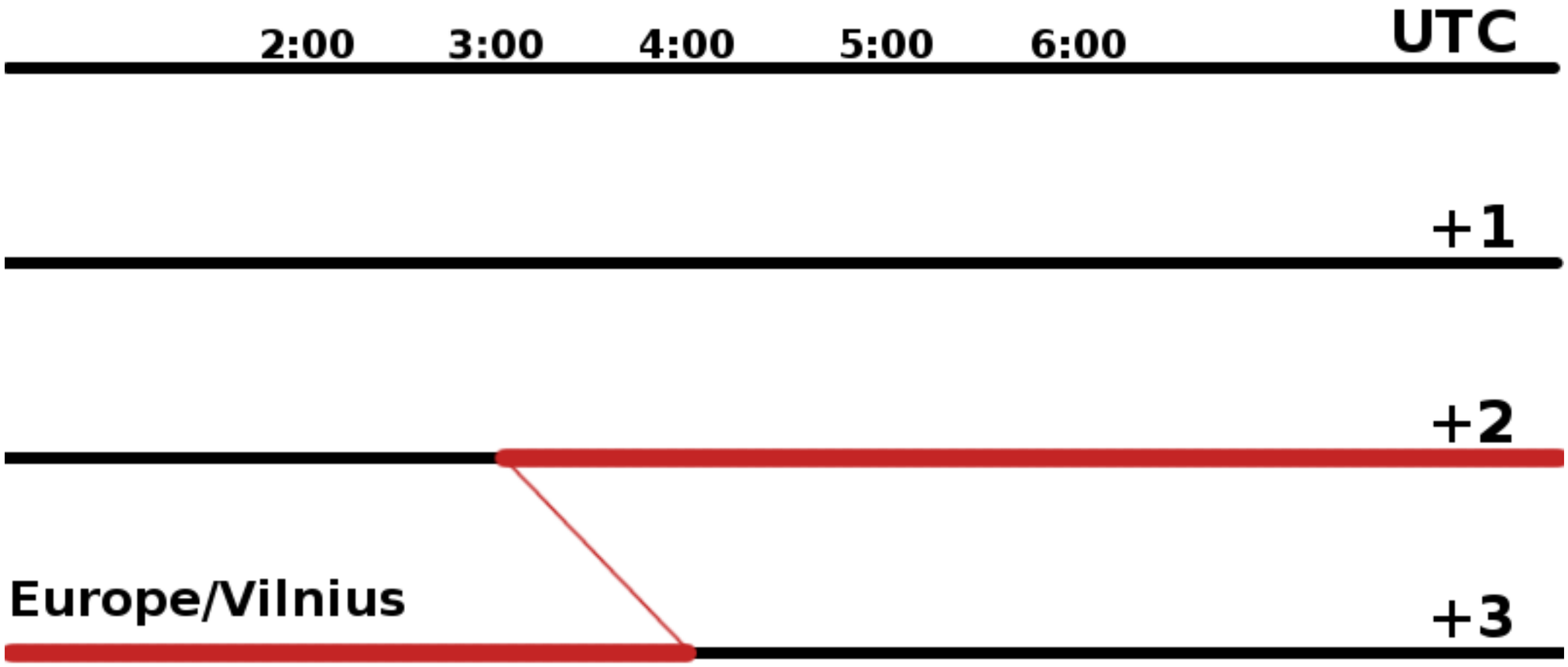
pytz

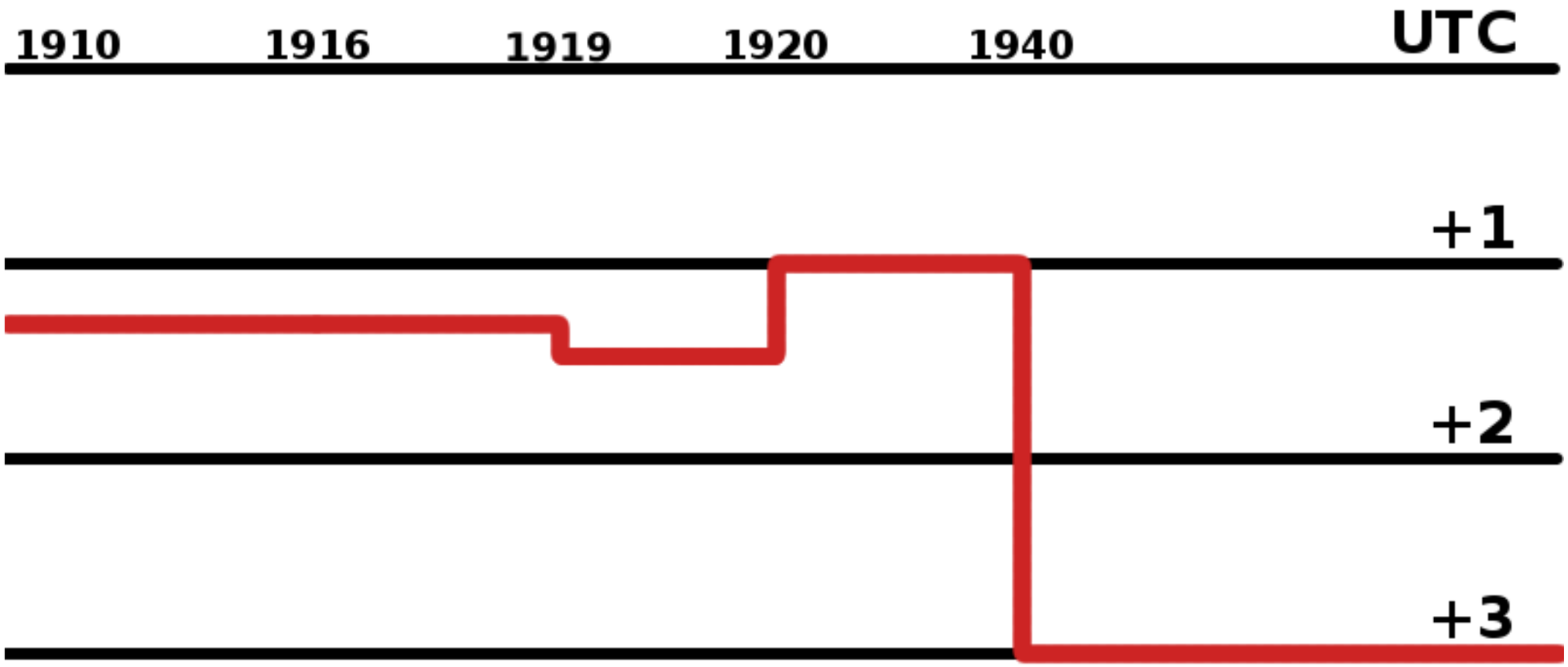

7:00
8:00
9:00
10:00
11:00
UTC
+1
+2
+3
Same timestamp
Same time on the clock

2006-03-26



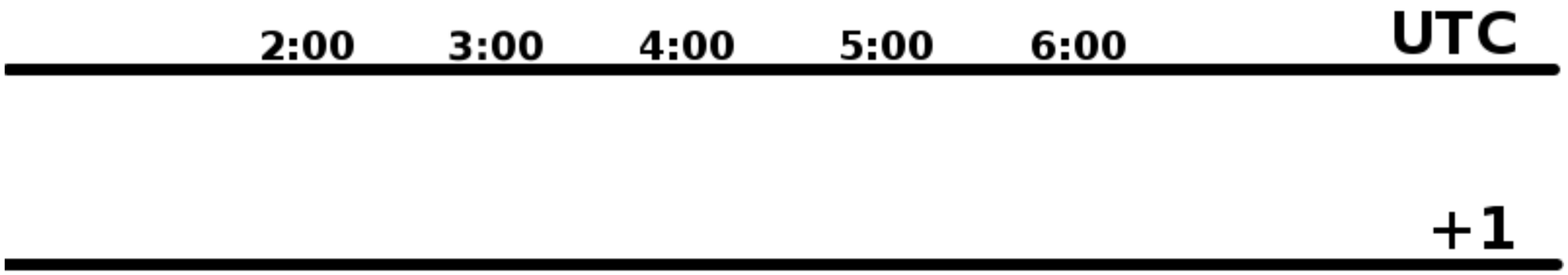
2006-10-29

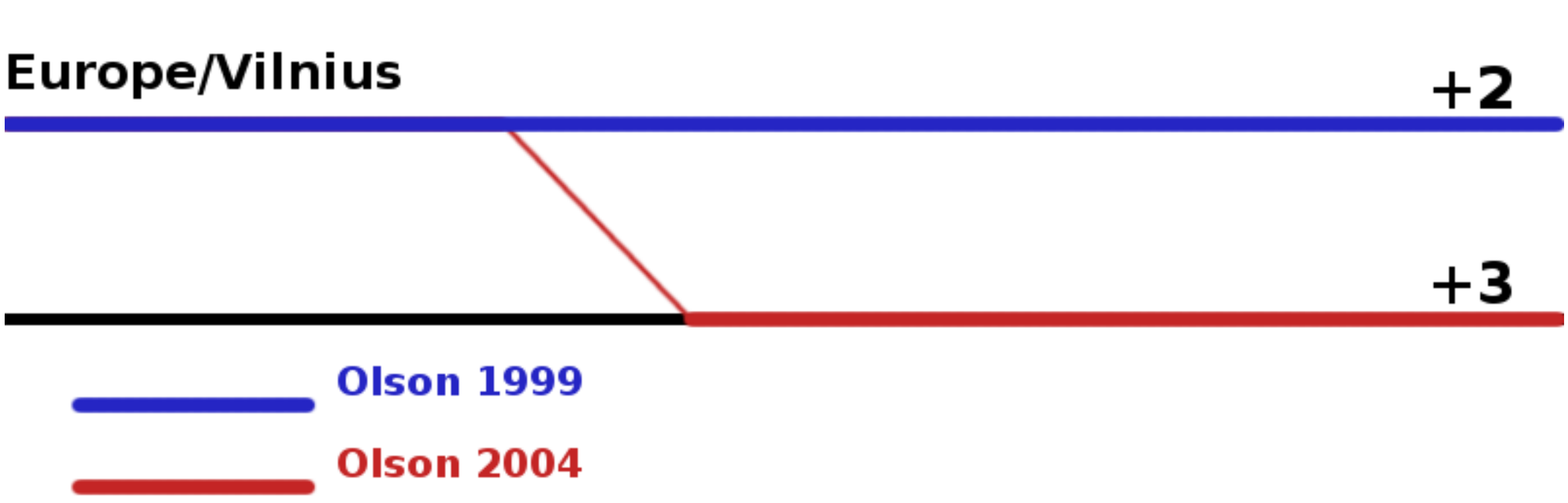



1910
1916
1919
1920
1940
UTC
+1
+2
+3

Olson.

2003-03-30


2:00
3:00
4:00
5:00
6:00
UTC
+1


Europe/Vilnius
+2
+3
Olson 1999
Olson 2004

Parsing user entered dates, creating new timezone aware timestamps.

These are wrong:

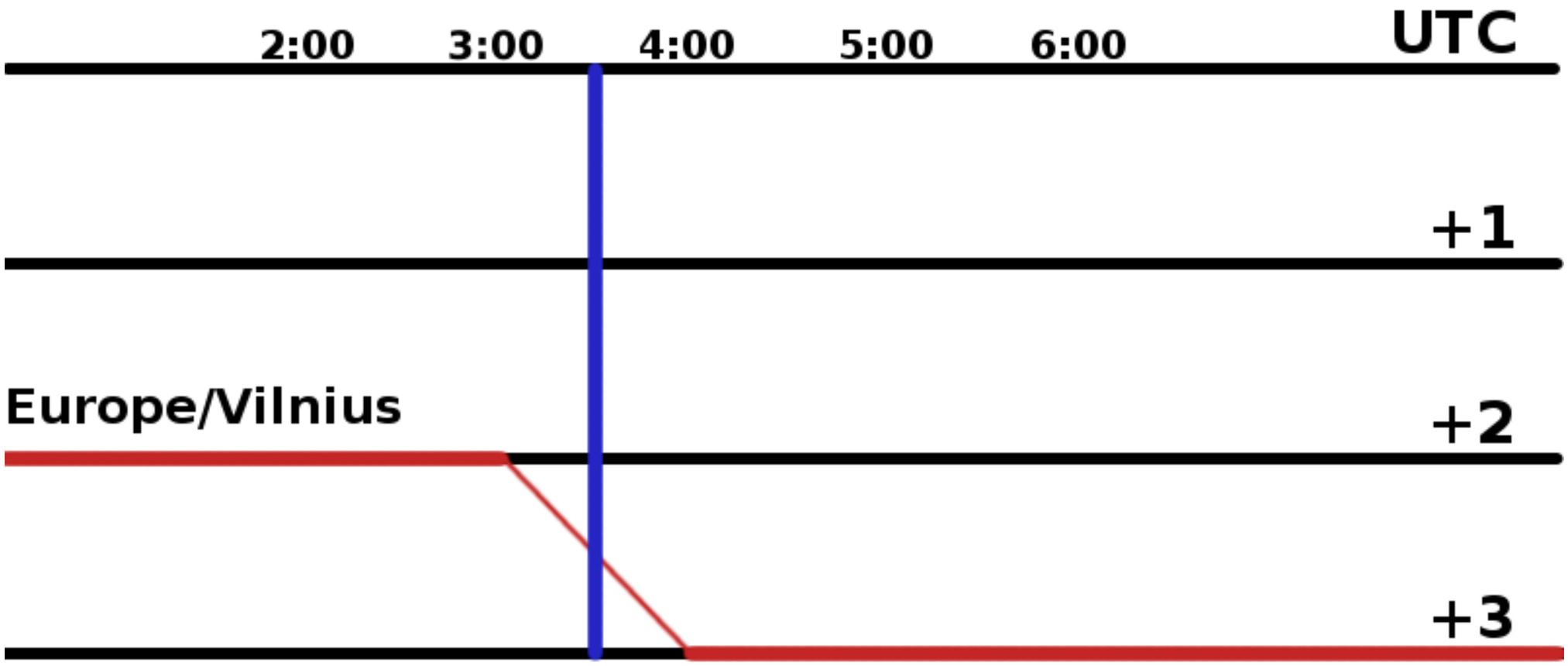
```
>>> tz = pytz.timezone('Europe/Vilnius')
>>> datetime(2006, 3, 26, 3, 30,
...          tzinfo=tz)
>>> datetime(2006, 10, 29, 3, 30,
...          tzinfo=tz)
```

But why?

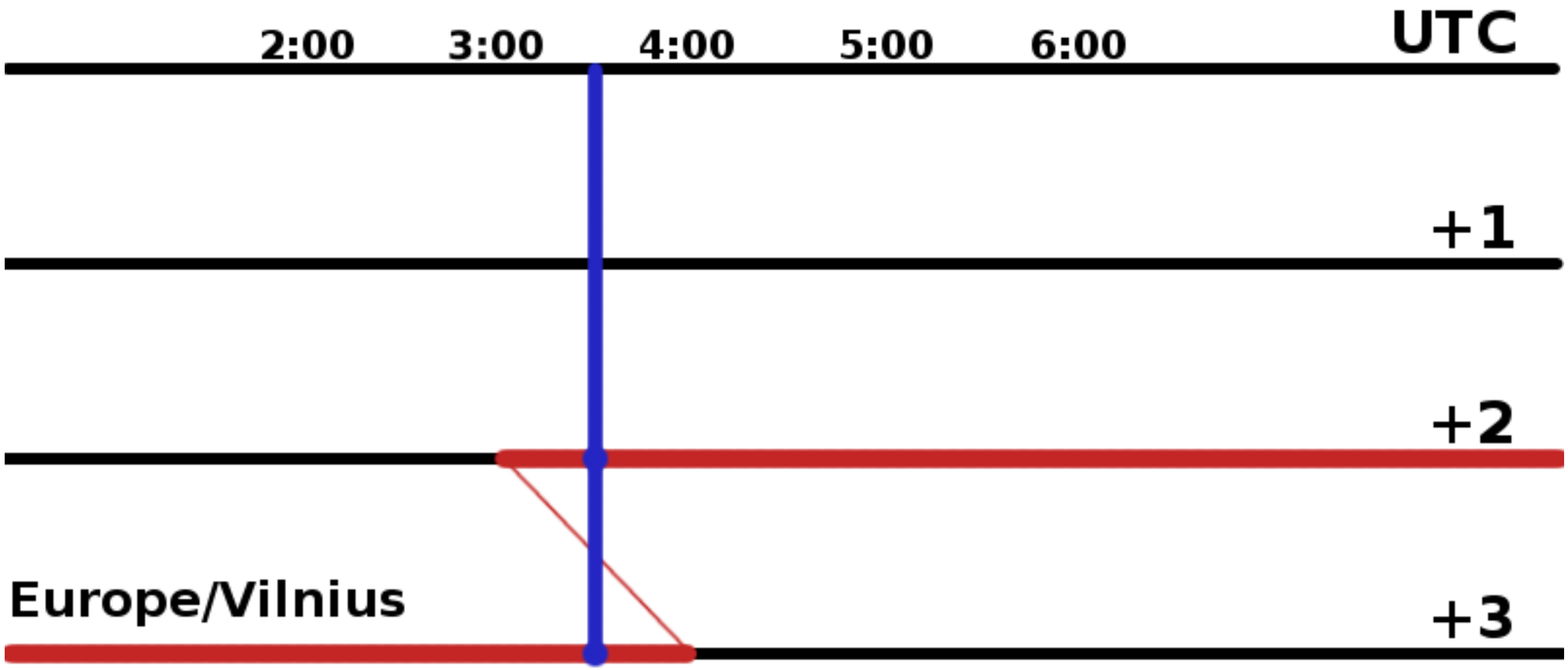
```
>>> tz = pytz.timezone('Europe/Vilnius')
>>> dt1 = datetime(2006, 3, 26, 3, 30,
...                tzinfo=tz)
>>> dt1.astimezone(pytz.UTC)
datetime.datetime(
    2006, 3, 26, 2, 6, tzinfo=<UTC>)
```

```
>>> dt2 = datetime(2006, 10, 29, 3, 30,
...                tzinfo=tz)
>>> dt2.astimezone(pytz.UTC)
datetime.datetime(
    2006, 10, 29, 2, 6, tzinfo=<UTC>)
```

2006-03-26



2006-10-29



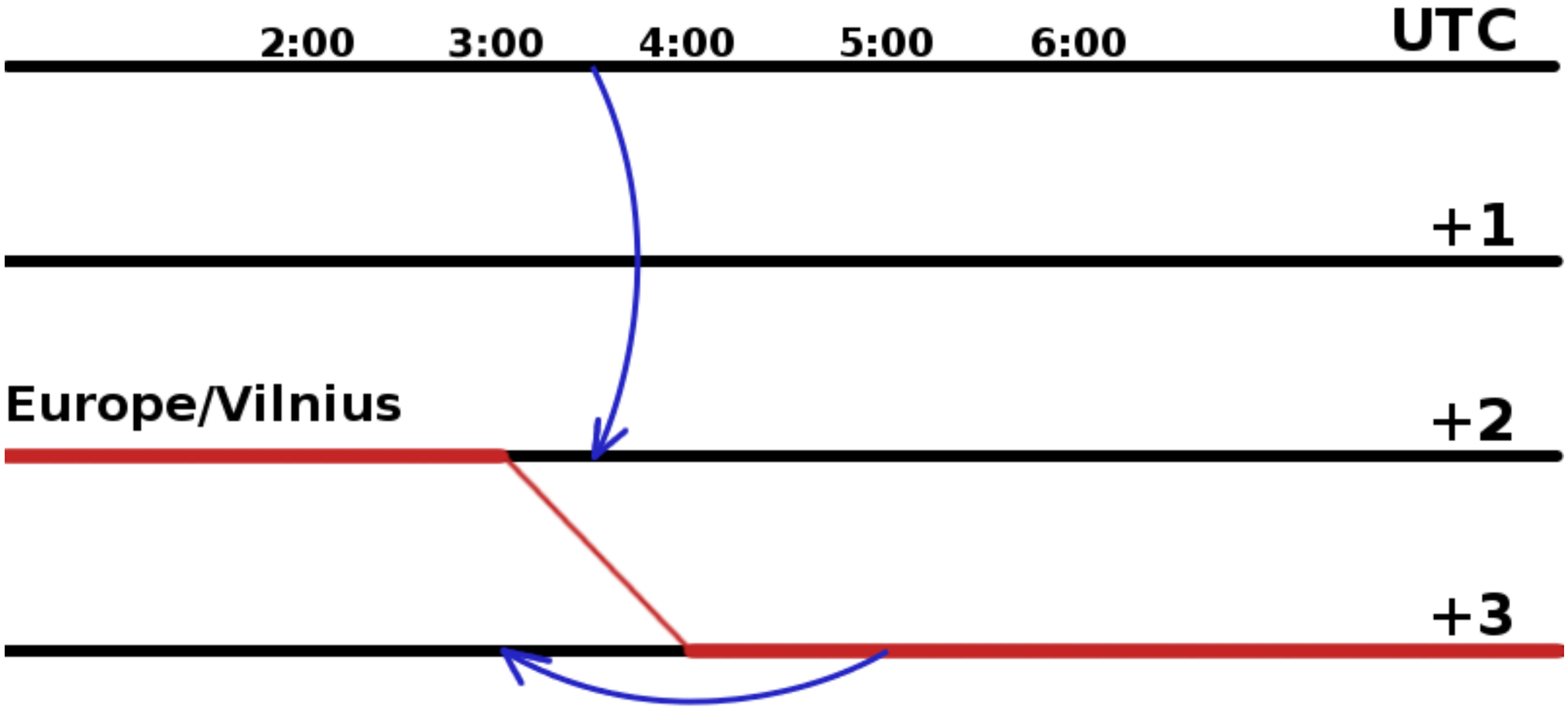
```
>>> tz = pytz.timezone('Europe/Vilnius')
>>> dt = datetime(2006, 3, 26, 3, 30)
>>> tz.localize(dt)
Traceback (most recent call last):
  . . .
IndexError: list index out of range
```

```
>>> dt = datetime(2006, 10, 29, 3, 30)
>>> tz.localize(dt, is_dst=True)
datetime.datetime(2006, 10, 29, 3, 30,
    tzinfo=<DstTzInfo 'Europe/Vilnius'
EET+2:00:00 STD>)
>>> tz.localize(dt, is_dst=False)
datetime.datetime(2006, 10, 29, 3, 30,
    tzinfo=<DstTzInfo 'Europe/Vilnius'
EEST+3:00:00 DST>)
```

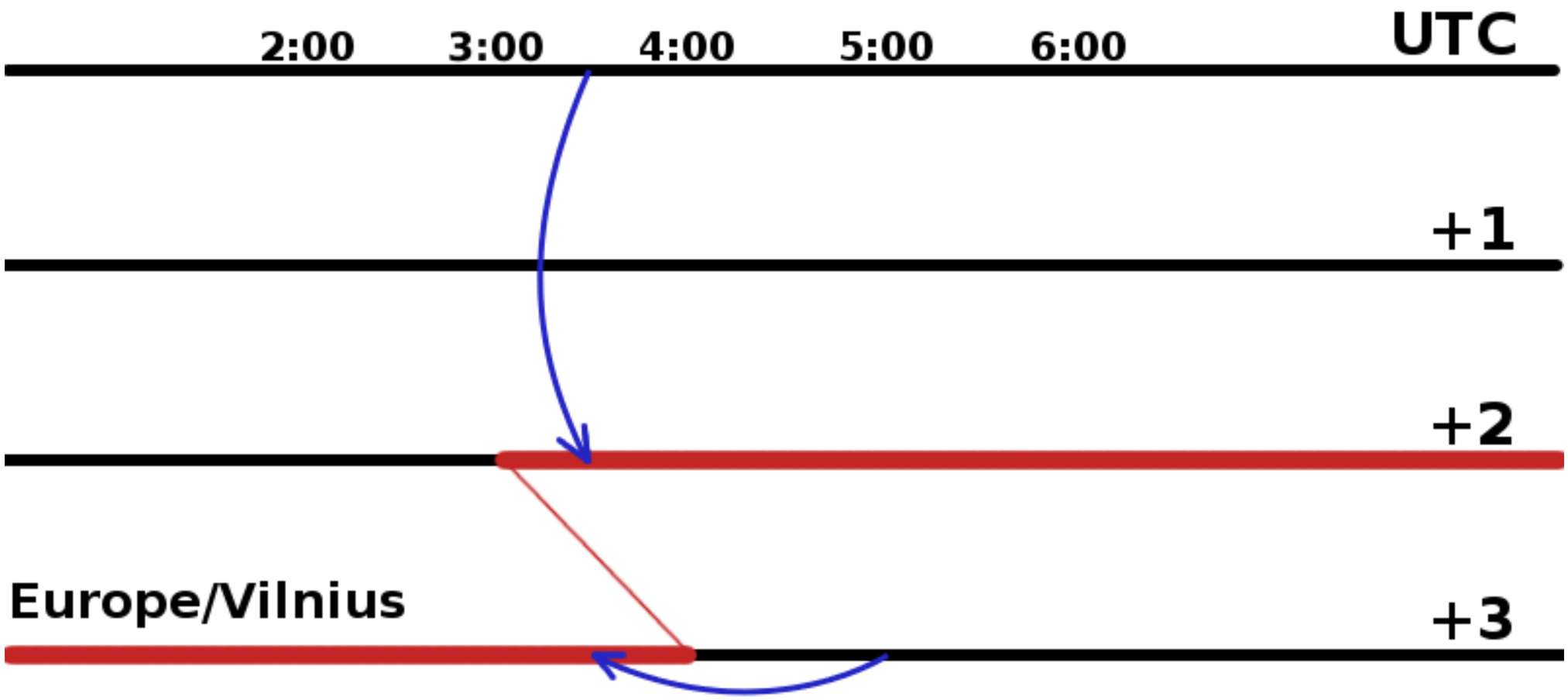
```
>>> dt = datetime(2006, 10, 29, 3, 30)
>>> tz.localize(dt, is_dst=None)
Traceback (most recent call last):
  ...
pytz.tzinfo.AmbiguousTimeError:
2006-10-29 03:30:00
```

Replace

2006-10-29



2006-10-29



Today

First one is simple:

```
>>> datetime.date.today()
datetime.date(2006, 7, 4)
```

Returns the date in the timezone of the server.

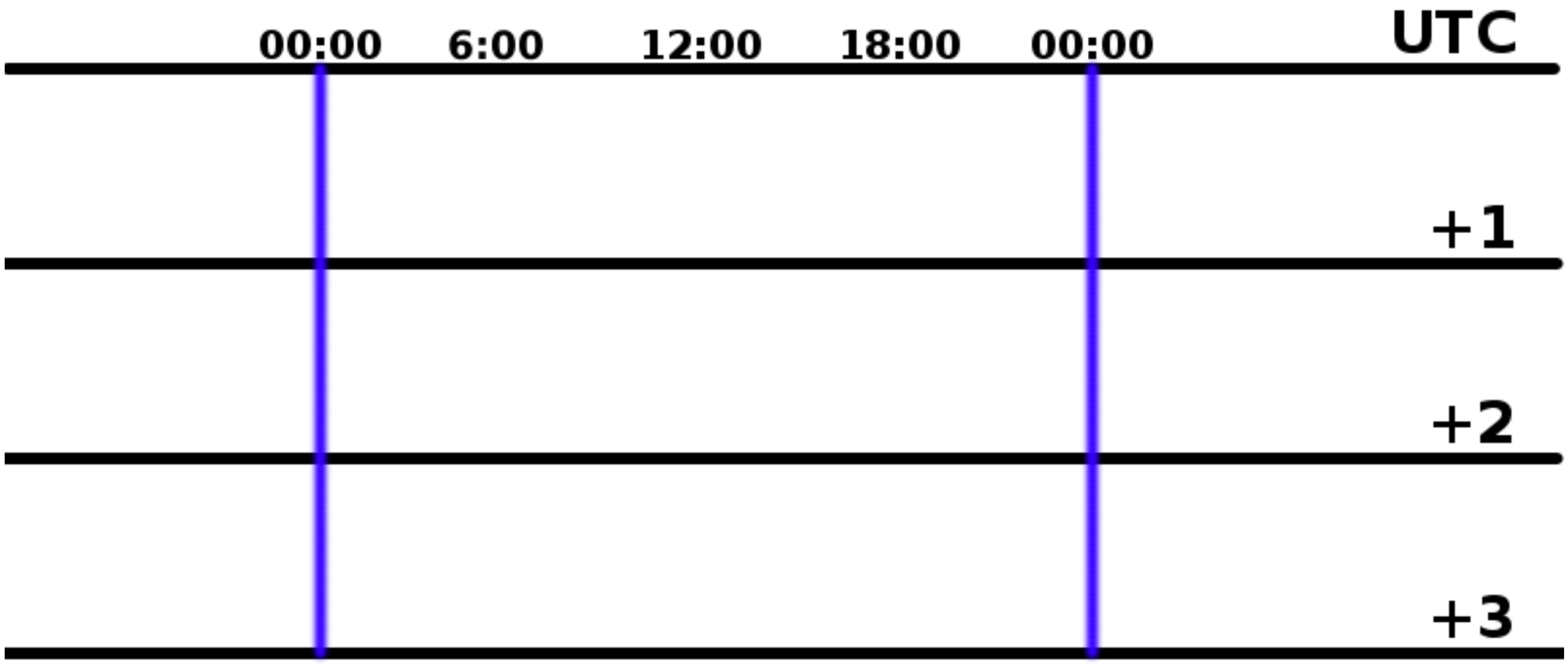
Do it this way:

```
>>> tz = pytz.timezone('Europe/Zurich')
>>> dt = utc.localize(datetime.utcnow())
>>> dt.astimezone(tz).date()
```

Dates are not timezone aware.

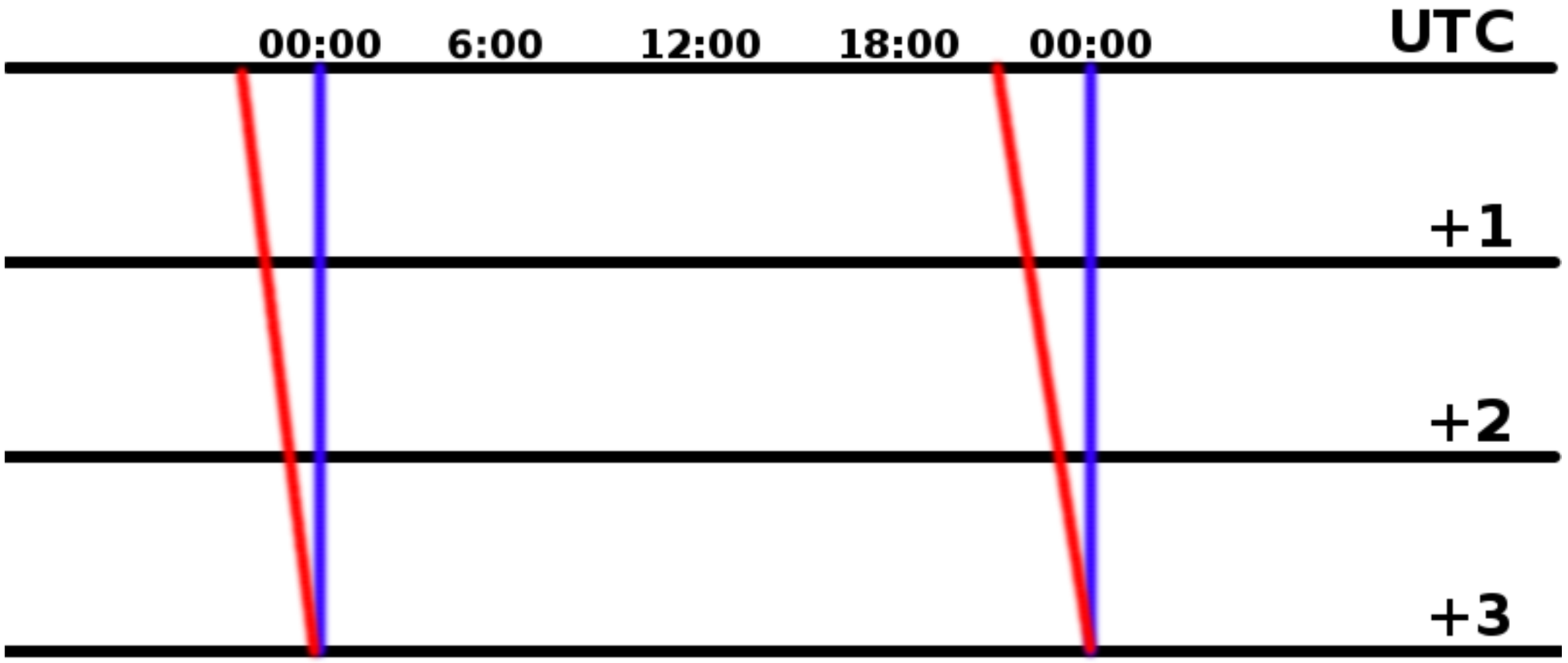
Comparing dates to other dates is easy.

Transforming dates to time intervals is not.


00:00
6:00
12:00
18:00
00:00
UTC
+1
+2
+3

Events that are happening on this particular date:

```
>>> [dt for dt in timestamps
...  if dt.date() == my_date]
```


00:00
6:00
12:00
18:00
00:00
UTC
+1
+2
+3

Events that are happening between 00:00 and 23:59:59.999999...

```
>>> [dt for dt in timestamps
...  if dt.astimezone(tz).date() ==
my_date]
```