Subject: timegen
Posted by spluque on Wed, 30 Oct 2013 20:56:35 GMT
View Forum Message <> Reply to Message

Hi,

I'm having a little difficulty generating time sequences with specific start and end values. Say, for instance we need to generate a sequence at 1 seconds and I need to cover the full final day. I thought this would do:

```
beg_jd=julday(10, 10, 2013, 0)
end_jd=julday(10, 11, 2013, 0)
step_size=0.1
ts=timegen(start=beg_jd, $
        final=end_jd + 1 - (float(step_size) / 86400), $
        step_size=step_size, units='seconds')
```

But, checking:

```
caldat, ts, mo, dd, yyyy, hh, mm, ss
print, yyyy[-1], mo[-1], dd[-1], hh[-1], mm[-1], ss[-1]
```

prints:

```
2013        10        11        23        59      59.802656
```

The last step is missing. What is wrong with this?

Thanks,
Seb

Subject: Re: timegen
Posted by Phillip Bitzer on Wed, 30 Oct 2013 22:16:57 GMT
View Forum Message <> Reply to Message

This may be pertinent, from the help:

Note: If the step size is not an integer then the last element may not be equal to the FINAL time. In this case, TIMEGEN will return enough elements such that the last element is less than or equal to FINAL.

If you 'caldat' your final time (end_jd + 1 - (float(step_size) / 86400)   ), I suspect this is the issue.

But, I'm not sure why this isn't what you're looking for:

```
ts=timegen(start=beg_jd, final=end_jd+1,step_size=step_size, units='seconds')
```

```
caldat, ts, mo, dd, yyyy, hh, mm, ss
print, yyyy[-1], mo[-1], dd[-1], hh[-1], mm[-1], ss[-1]
       2013          10          11          23          59       59.902636
```

BTW, your original code contains the line
float(step_size)
which is superfluous - step_size is already a float, yes?

Subject: Re: timegen
Posted by suicidaleggroll on Wed, 30 Oct 2013 22:38:01 GMT

View Forum Message <> Reply to Message

On Wednesday, October 30, 2013 2:56:35 PM UTC-6, spl...@gmail.com wrote:
> Hi,
>
>
>
> I'm having a little difficulty generating time sequences with specific start and end values. Say, for instance we need to generate a sequence at 1 seconds and I need to cover the full final day. I thought this would do:
>
>
>
> beg_jd=julday(10, 10, 2013, 0)
>
> end_jd=julday(10, 11, 2013, 0)
>
> step_size=0.1
>
> ts=timegen(start=beg_jd, $
>
>   final=end_jd + 1 - (float(step_size) / 86400), $
>
>   step_size=step_size, units='seconds')
>
>
>
> But, checking:
>
>
>
> caldat, ts, mo, dd, yyyy, hh, mm, ss
>
> print, yyyy[-1], mo[-1], dd[-1], hh[-1], mm[-1], ss[-1]
>
>
>

> prints:
>
>
>
> 2013   10   11   23   59   59.802656
>
>
>
> The last step is missing. What is wrong with this?
>
>
>
> Thanks,
>
> Seb

Never used timegen...what's so difficult about just doing:

```
beg_jd=julday(10,10,2013,0)
end_jd=julday(10,11,2013,0)
step_size=0.1/86400d0
 ts=dindgen(round((end_jd-beg_jd)/step_size)+1)*step_size+beg _jd
```

Subject: Re: timegen
Posted by spluque on Thu, 31 Oct 2013 02:36:44 GMT
View Forum Message <> Reply to Message

On Wed, 30 Oct 2013 15:16:57 -0700 (PDT),
Phillip Bitzer <bitzerp@uah.edu> wrote:

> This may be pertinent, from the help: Note: If the step size is not an
> integer then the last element may not be equal to the FINAL time. In
> this case, TIMEGEN will return enough elements such that the last
> element is less than or equal to FINAL.

> If you 'caldat' your final time (end_jd + 1 - (float(step_size) /
> 86400)), I suspect this is the issue.

> But, I'm not sure why this isn't what you're looking for:

> ts=timegen(start=beg_jd, final=end_jd+1,step_size=step_size,
> units='seconds') caldat, ts, mo, dd, yyyy, hh, mm, ss print, yyyy[-1],
> mo[-1], dd[-1], hh[-1], mm[-1], ss[-1] 2013 10 11 23 59 59.902636

> BTW, your original code contains the line float(step_size) which is
> superfluous - step_size is already a float, yes?

Thanks very much for these pointers. I left the float() call with step_size by accident here. In the actual code, this is part of a procedure that needs to be quite general for any step_size, so I'm coercing it to float. You're quite right about the note in the help page; that's exactly what's going on. To protect against this, the following seems to do what I need:

```
  step_d=float(step_time) / 86400
  times=timegen(start=beg_jd, $
                final=end_jd + 1 - (step_d / 2), $
                step_size=step_time, units='seconds')
```

i.e. adding half a step to 'final' ensures that the last step is always included.

Thanks,

--
Seb