

```
In [1]: import pandas as pd
store = pd.HDFStore('/Volumes/FreshBooks/data/store.h5')
may07 = store['may07']
```

If we want to get a graph over time like those you get from munin and graphite, we'll need to `resample()` our DataFrame:

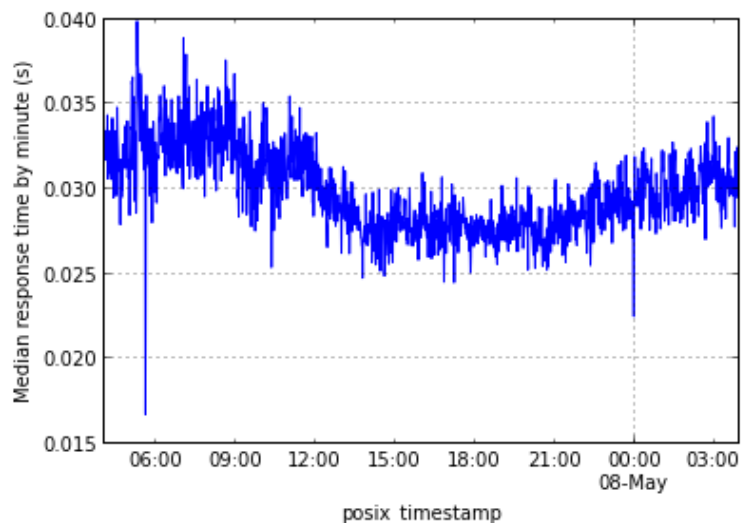
```
In [2]: from datetime import timedelta
import numpy as np
first_minutes = may07[
    may07.index < (may07.index[0] + timedelta(minutes=10))]
first_minutes['elapsed'].resample('T', how=np.median)
```

```
Out[2]: posix_timestamp
2012-05-07 04:00:00    0.033588
2012-05-07 04:01:00    0.032231
2012-05-07 04:02:00    0.031700
2012-05-07 04:03:00    0.031906
2012-05-07 04:04:00    0.033401
2012-05-07 04:05:00    0.032794
2012-05-07 04:06:00    0.031914
2012-05-07 04:07:00    0.031836
2012-05-07 04:08:00    0.034356
2012-05-07 04:09:00    0.030676
Freq: T, dtype: float64
```

Then we can use the handy `DataFrame.plot()` method to get it displayed easily:

```
In [3]: ylabel('Median response time by minute (s)')
may07['elapsed'].resample('T', how='median').plot()
```

```
Out[3]: <matplotlib.axes.AxesSubplot at 0x10587fa10>
```



We can use other callables to indicate how we'd like the downsampling to occur, such as a percentile function:

```
In [4]: elapsed = may07['elapsed']
ylabel('Seconds')
elapsed.resample(
    'T', how=lambda x: np.percentile(x, 25)
).plot(label="25%ile", color='red')
elapsed.resample(
    'T', how=lambda x: np.percentile(x, 50)
).plot(label="50%ile", color='orange')
elapsed.resample(
    'T', how=lambda x: np.percentile(x, 75)
).plot(label="75%ile", color='green')
legend()
```

Out[4]: <matplotlib.legend.Legend at 0x111b46e50>

