Setup

```
using Plots
```

```
0.002
```

```
begin
    T = 1
    L = 500
    Δt = T/L
end
```

```
ΔW =
 [-0.0364173, -0.0417868, -0.0517011, -0.0713592, 0.0108349, 0.031157, -0.0103997, 0.0471(
```

```
ΔW = sqrt(Δt)*randn(L)
```

```
h′ (generic function with 1 method)
```

```
begin #Define the functions of interest
    h(x) = x.^3
    h′(x) = 3 * x.^2
end
```

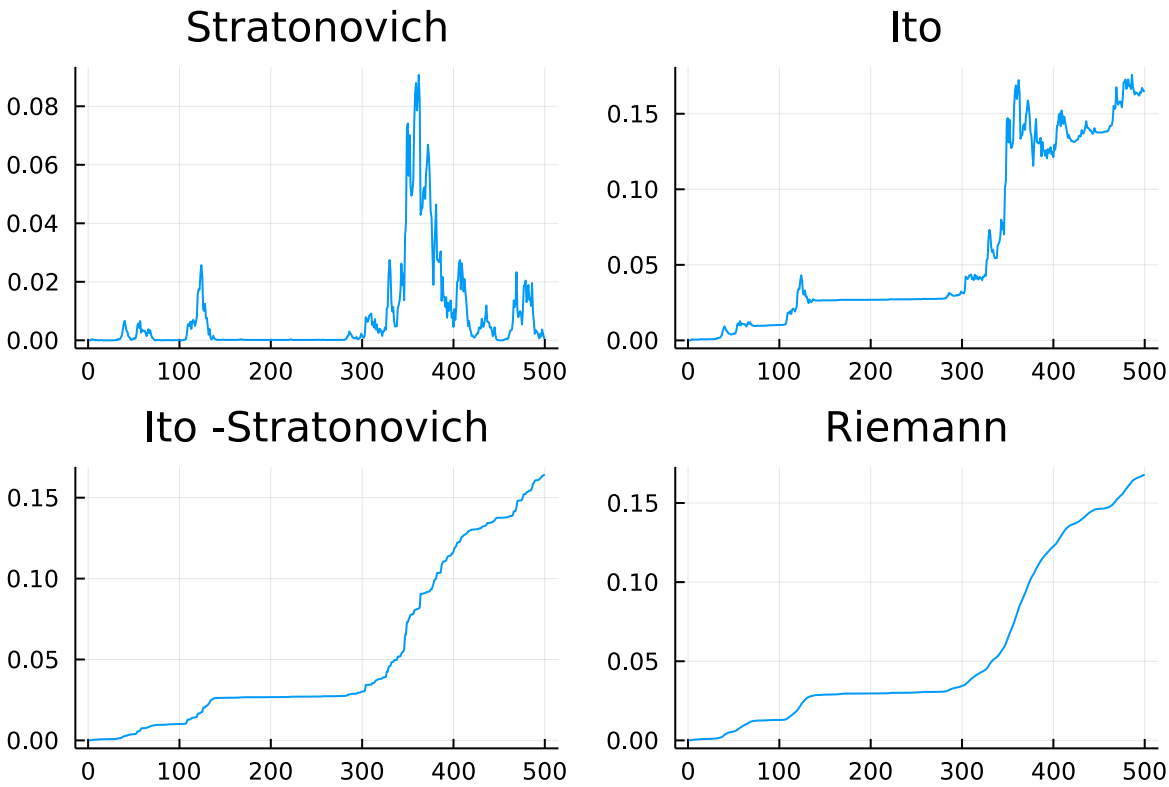
```
generate_plots (generic function with 1 method)
```

```
begin
    function get_path(ΔW::Array, h::Function, h′::Function)
        #setup processes
        W = cumsum(ΔW)
        midpoints(W) = ([0, W[1:end-1]...] + W)./2

        #calculate integrals
        stratonovich = cumsum(h(midpoints(W)).*ΔW)
        ito = cumsum(h(W).*ΔW)
        riemann = cumsum(h′(W))*Δt/2

        #return
        return (stratonovich=stratonovich, ito=ito, riemann=riemann)
    end

    function calculate_msr(nt)
        return cumsum((nt.ito - nt.stratonovich - nt.riemann).^2)
    end

    function generate_plots(nt)
        #Plot Integrals
        p1 = plot(nt.stratonovich, title="Stratonovich");
```

Problem P7.1

```
paths =
  (stratonovich = [2.19857e-7, 8.0857e-6, 6.6334e-5, 0.000390308, 0.000308918, 0.000142362
```

```
paths = get_path(ΔW,h,h′)
```

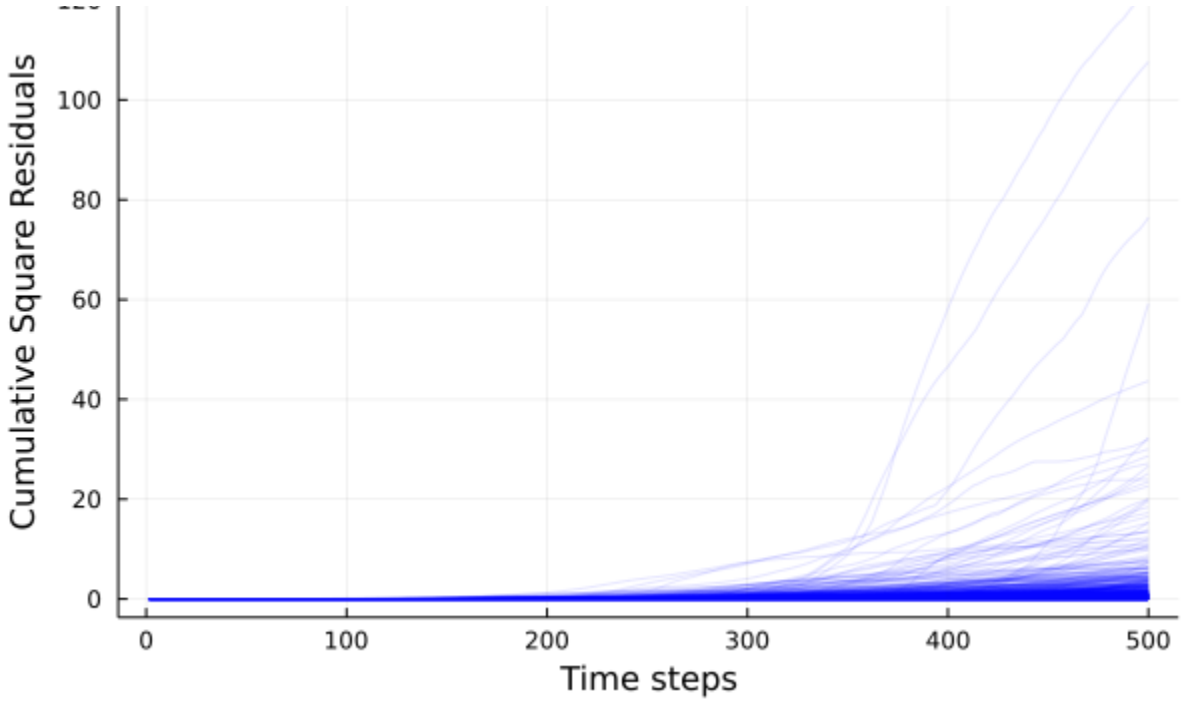



```
generate_plots(paths)
```

Problem P7.2

The way I decided to test how well this worked was to measure the accumulated square residuals to see how badly things diverged. In general, things line up as they should.

```
begin
    N=1500
```




```
plot(p,color=:blue,alpha=0.1
    ,legend=false
    ,ylabel="Cumulative Square Residuals"
    ,xlabel="Time steps"
    ,title="Accumulation of Squared Residuals"
)
```