Dean Katsaros
Homework 3

The multiscale SDE

$$\frac{dx}{dt} = (1 - y^2)x,$$

$$\frac{dy}{dt} = -\frac{\alpha}{\epsilon}y + \sqrt{\frac{2\alpha}{\epsilon}}\frac{dW}{dt}$$

where $W(t)$ is a one dimensional Brownian motion. Setting $\lambda = 1.1$, $\alpha = 1$, the theory presented in Chapter 11 says that the effective equations are $\frac{dX}{dt} = F(x) \cdot \nabla_x X(x,t) + \frac{1}{2}A(x)A(x)^{:}\nabla_x \nabla_x X(x,y)$, with $A(X)A(X)^T$, which translates to the equation $\frac{dX}{dt} = \left(1 - \frac{\lambda}{\alpha}\right)X$, and thus the effective averaged equation is $X = X_0 e^{\left(1-\frac{\lambda}{\alpha}\right)t}$. We will assume an initial condition $X_0 = 1$, and thus compare all numerical method results to plots of $X(t) = e^{\left(1-\frac{\lambda}{\alpha}\right)t}$.

A note: All plots will include 3 values of Dt, each smaller than the last. These graphs are plotted against time, and on the same plot is a (dark blue and bold) graph of the $X(t)$ above. Errors at the finest mesh (largest Dt) are also plotted below the graphs of the integrated solutions.

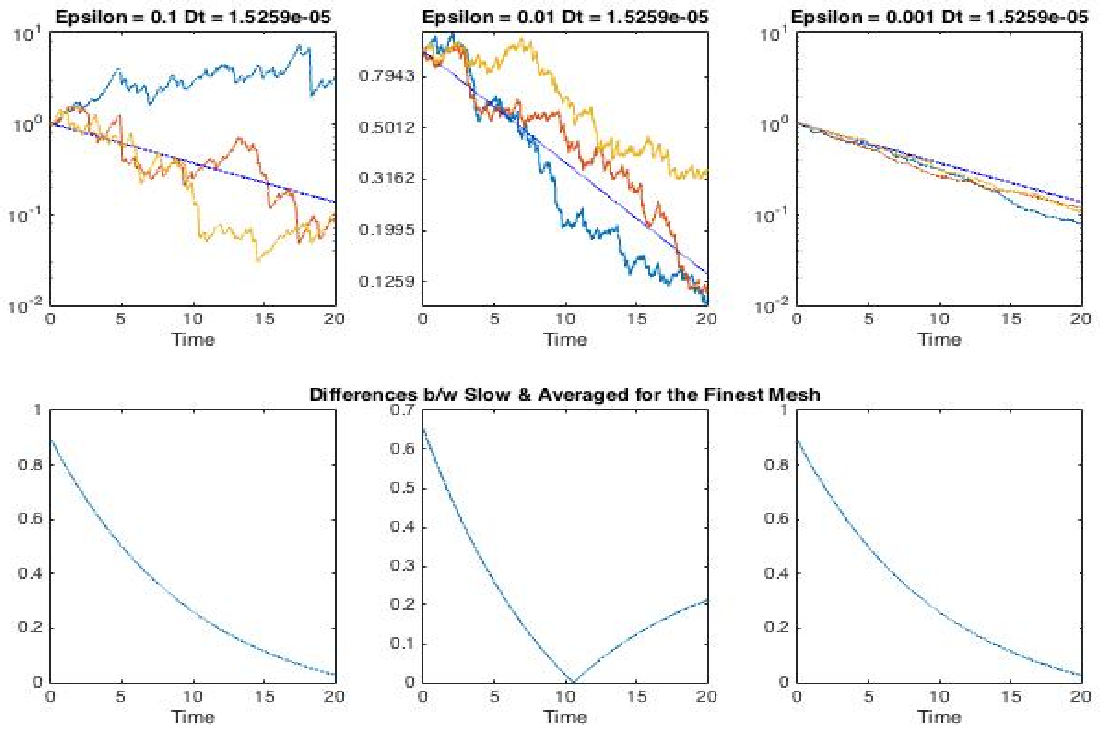
For all three simulations, the Euler -Maruyama Numerical Integration Method was utilized to integrate the SDE in question. That is, for an SDE $dX = a(X)dt + b(X)dW$, modify the Euler Method to include a term representing the brownian motion $W(t)$, to get the scheme

$$Y_{n+1} = Y_n + a(Y_n)\Delta t + b(Y_n)\Delta W_n,$$

where Δt is a time step in a $[0, T]$ mesh, and ΔW_n is a sample of the brownian motion. Formally, $\Delta t = T/N$ where N is the mesh size, $\{\tau_n \in [0,T]\}_{n=0}^{N}$, and $\Delta W_n = W_{\tau_{n+1}} - W_{\tau_n}$. In our case, W_n will be one of N random variables from the standard normal distribution.

Direct Integration.

The Euler- Maruyama integration method was used to calculate the solution to $\frac{dx}{dt}$. This was then plotted and compared to a plot of $X(t) = e^{\left(1-\frac{\lambda}{\alpha}\right)t}$.



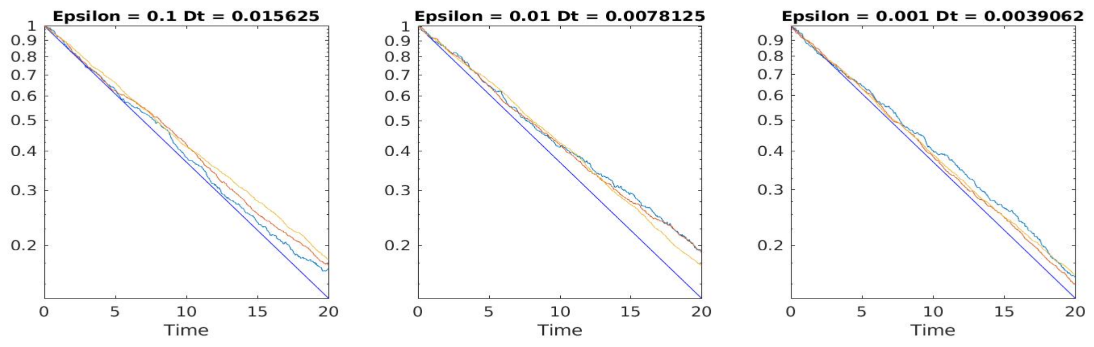
Forced Averaging Utilizing equation 10.5.3,

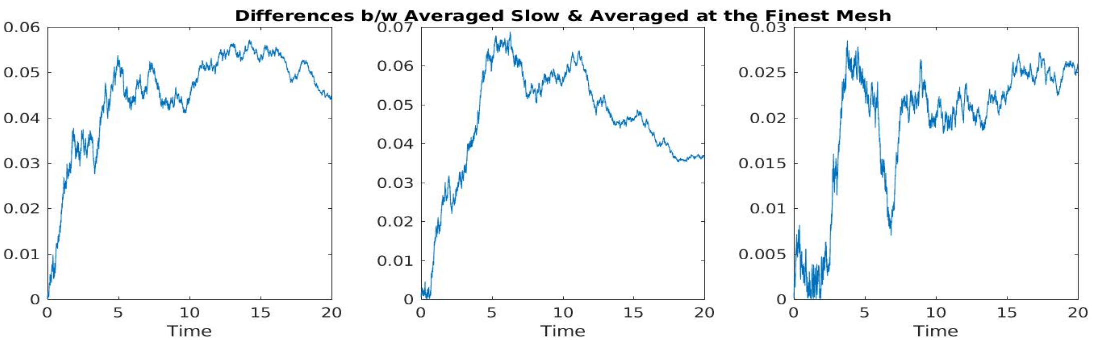
$$F(x) = \frac{1}{T}\int_0^T f(x, \varphi_x^s(y))ds,$$

where $\varphi_x^s(y)$ will be calculated by using Euler-Maruyama to integrate $\frac{dy}{dt}$. Then, this was plugged in for y in $\frac{dx}{dt}$, and Euler's method was used to integrate this equation, and then an average was taken over the solutions X.

```
N = T/Dt
L = eps / Dt2 (Comment) Fast time scale.
for j = 1:N
(Comment) Create $\phi_x^s$ some solution trajectory for the fast
(Comment) dynamics
Winc2 = sqrt(Dt2) * randn(M,L);
Ytemp = Yzero*ones(M,1);
for l = 1:L
Ytemp = Ytemp + Dt2*(-alpha/eps)*Ytemp+sqrt(2*lambda/eps).*Winc2(:,l);
end
(Comment) Integration and averaging of slow dynamics according to eqn 10.5.3
Xtemp = Xtemp+(1-Ytemp.^2).*Xtemp*Dt;
Xavg(j) = mean(Xtemp);
end
```

By examining the noise, it was clear that this was a more accurate , and consistent method for inetegrating the equations.





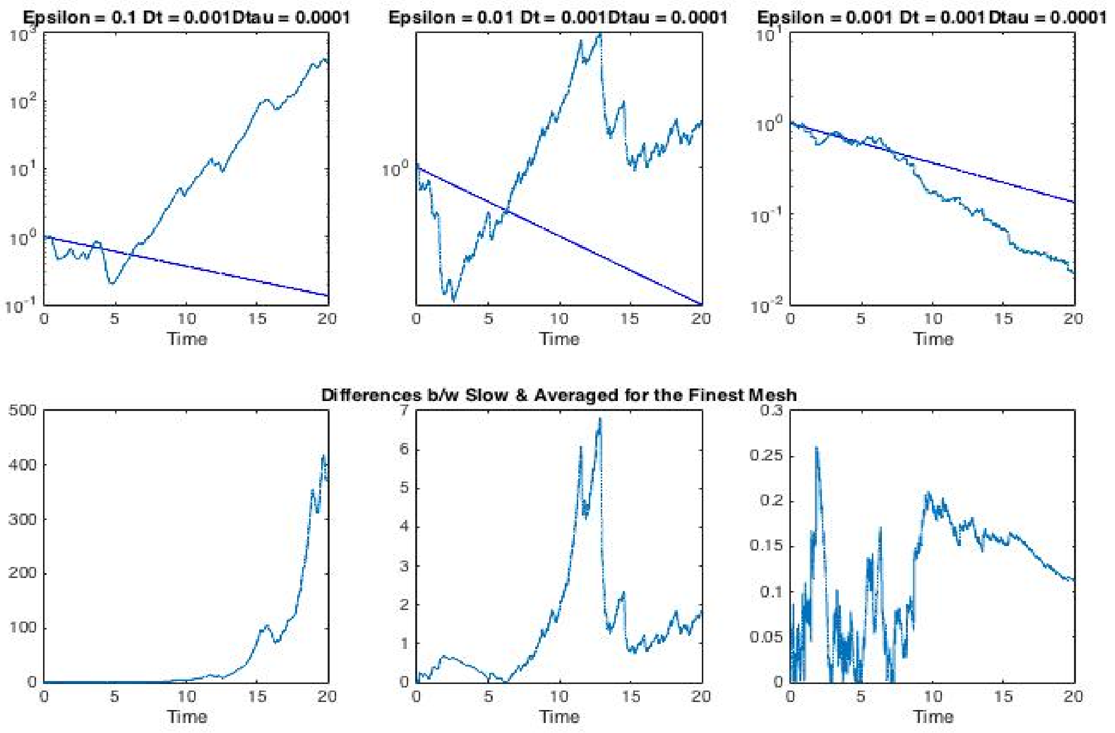
FLAVORS. The integration was conducted as follows.

a) A one step was calculated in both the fast and slow variables with a fast / small time step.

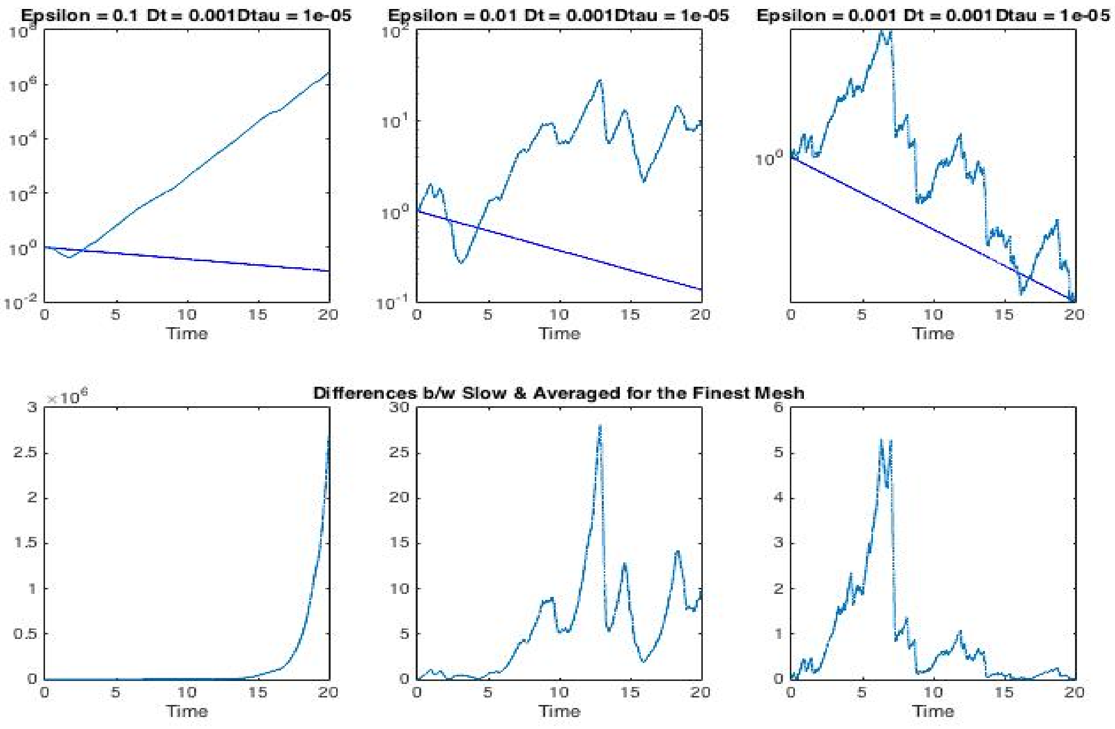
b) This was used to calculate one large time step of the slow variable.

Thus, effectively, two mesh sizes were utilized, one that represented the fast time scale, and one that represented the slow time scale (similar to forced averaging), but the fast time scale was utilized to calculate the slow variable and the fast variable, then the slow variable was calculated on a large time scale as a function of the previous results. This was then repeated, etc.

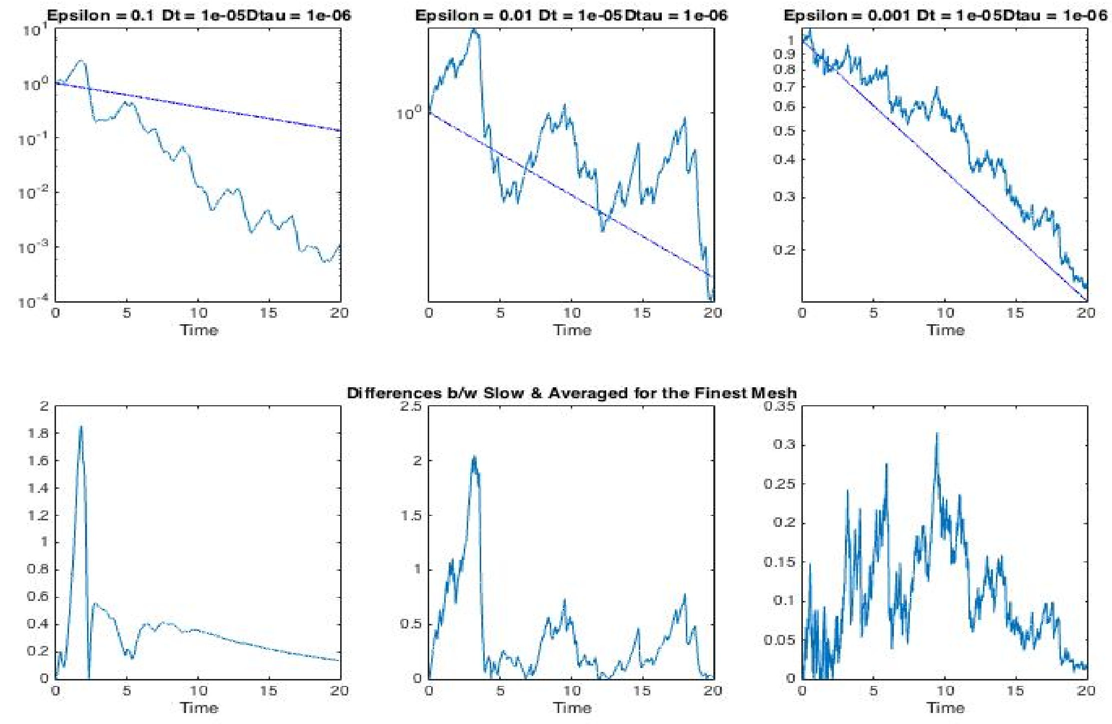
The graph of this and the errors shows that it was a poor approximation of the averaged dynamics.



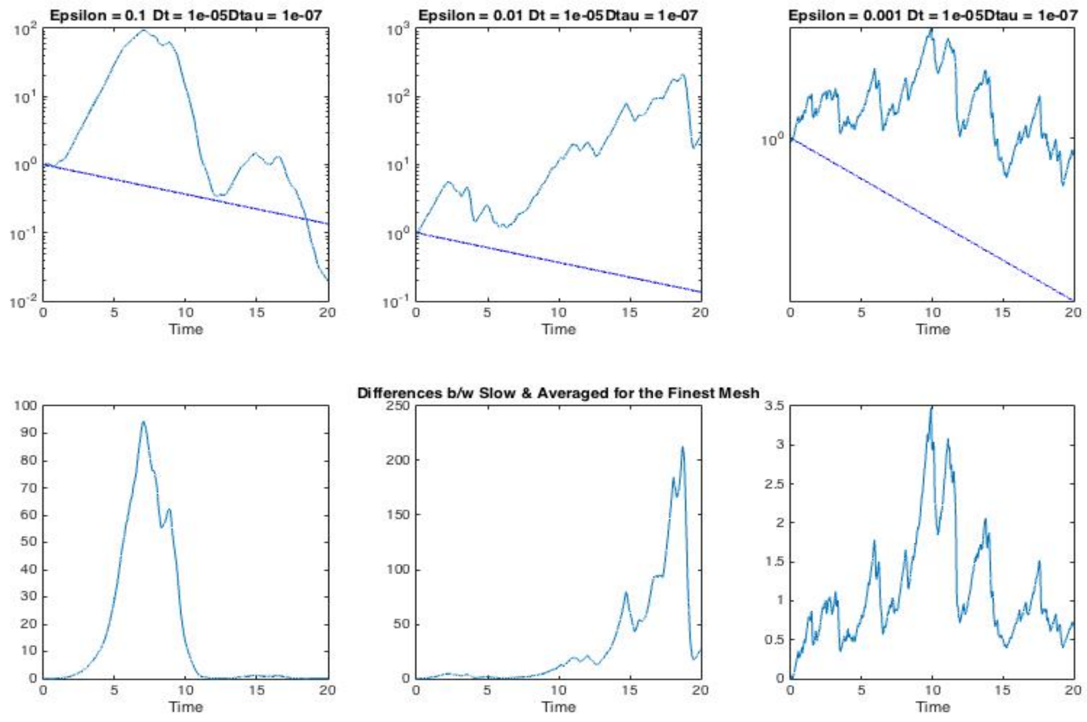
An interesting sidenote is that using a finer mesh size for the fast variable actually resulted in larger errors for this method.



If the mesh size for both the fast and slow are increased, but kept within a factor of 10 of one another, errors did improve.

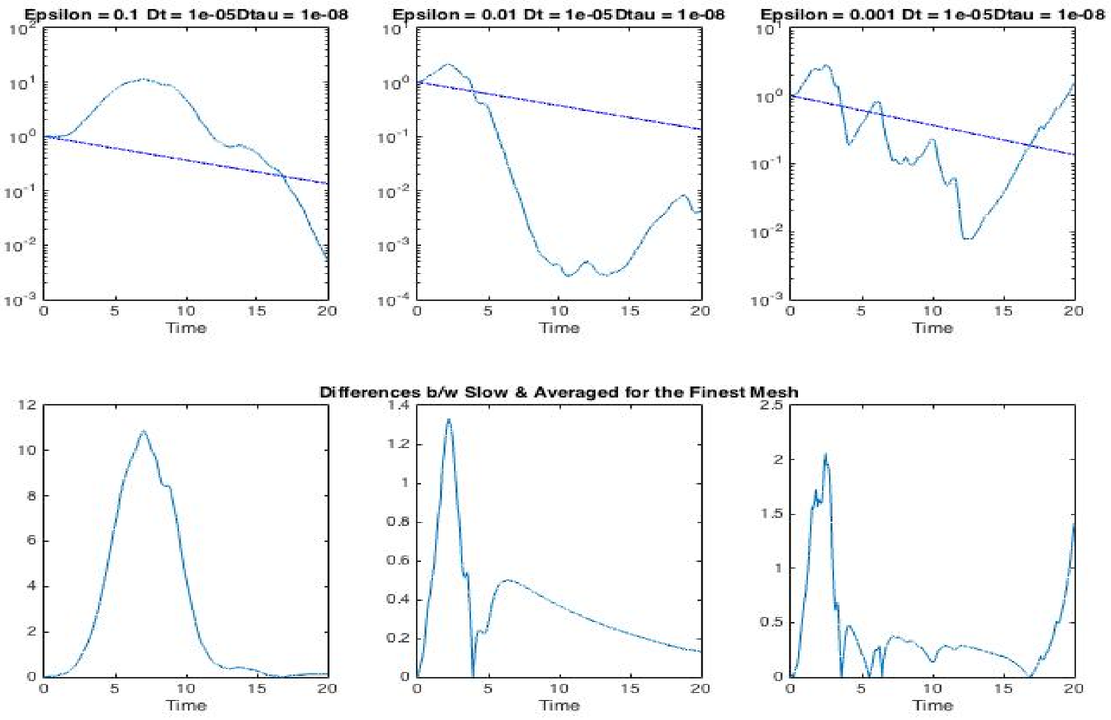


However, if $D\tau$ was changed to 10^{-7}, we, again, get a worse result.



This hints that the mesh sizes for the fast and slow variables cannot be too different, which is suspicious when considering the viability of FLAVORS as a numerical method for approximating the effective dynamics of an SDE.

The study was taken one step further by decreasing $D\tau$ by another factor of 10. This was actually more accurate then for $D\tau = 10^{-7}$, which is interesting to say the least.



FLAVORS is apparently somewhat unpredictable as far as its accuracy is concerned, at least in this implementation, and requires experimenting with the mesh sizes. Again, this makes it seem like a strange method, as the other methods utilized had a more direct relationship between the mesh sizes and the accuracies.