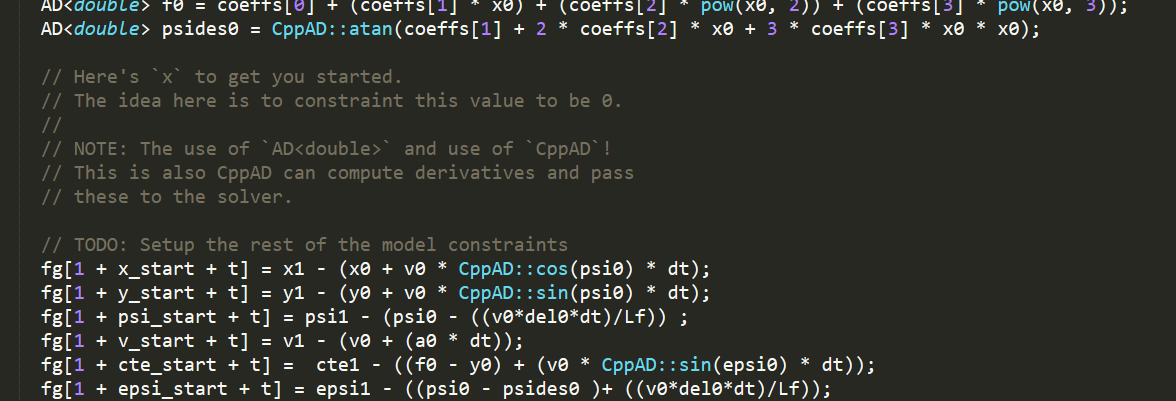
The Model

* Variables
  + X, Y, PSI (ANGLE), Speed V, CTE (cross track error), epsi (Error in psi)
  + Actuator inputs are
    - Steer value
    - Throttle value
  + You have equations governing the STATE(Previous) and STATE(Next)
  + Reference points are given and u got to find the predicted trajectory as close to that.
  + So, it becomes a Constrained optimization problem.
  + Constrained equations are (some of them)



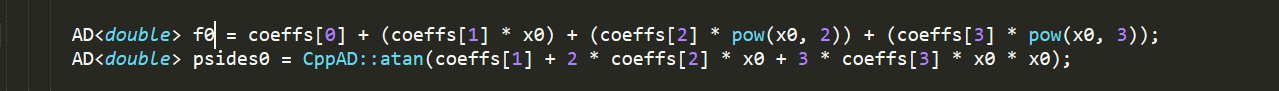
* Unique things I did
  + I have kept a cost for the deviation of speed from the reference velocity.
  + Have given extra cost for CTE and EPSI also have given more cost to extra deviations in the steering angles.
  + Have used a 3rd order polynomial for fitting the wave points.

Timestep Length and Elapsed Duration (N & dt)

* I just tried manually a lot of lot variations of “dt” and “N”.
  + For first Submission
  + N is 10
  + Dt is 1
    - Was chosen as my first submission and I was using single order polynomial to fit it.
    - Lot of deviations were there and there fore my project submission was rejected.
* Then in this submission I have given
  + N is 10
  + Dt is 0.3
    - Were empirically giving better results.
    - With dt greater then 1 or close to 1. The trajectory was going too far and that was not needed.
    - Also, when I plotted the green and yellow lines giving extra N does not make extra sense (for dt 0.3).

Polynomial Fitting and MPC Pre-processing

* Fitted a third order polynomial
  + Have fitted line to the points converted to the car coordinate.
  + auto coeffs = polyfit(x\_p, y\_p, 3) ;
  + also, in MPC.cpp you must use the coefficients to do the Job.



Model Predictive Control with Latency

* As there is latency, vehicle will be in a different position when u apply the controls. So, the assumption is let’s apply the controls on the state after the latency.
* So, after latency.
  + X is “v \* latency”
  + Psi is” -v \* steer angle \* latency / 2.67”.
  + Y is 0.0
  + cte is polyeval(coeffs, x\_latency) (I have not used latency here because it is still giving me good results)
  + epsi is -atan(coeffs[1] + 2 \* x\_latency \* coeffs[2] + 3 \* x\_latency \* x\_latency \* coeffs[3]); (I have not used latency here because it is still giving me good results)