2.7 Compute the probability density function (PDF) of *ST*

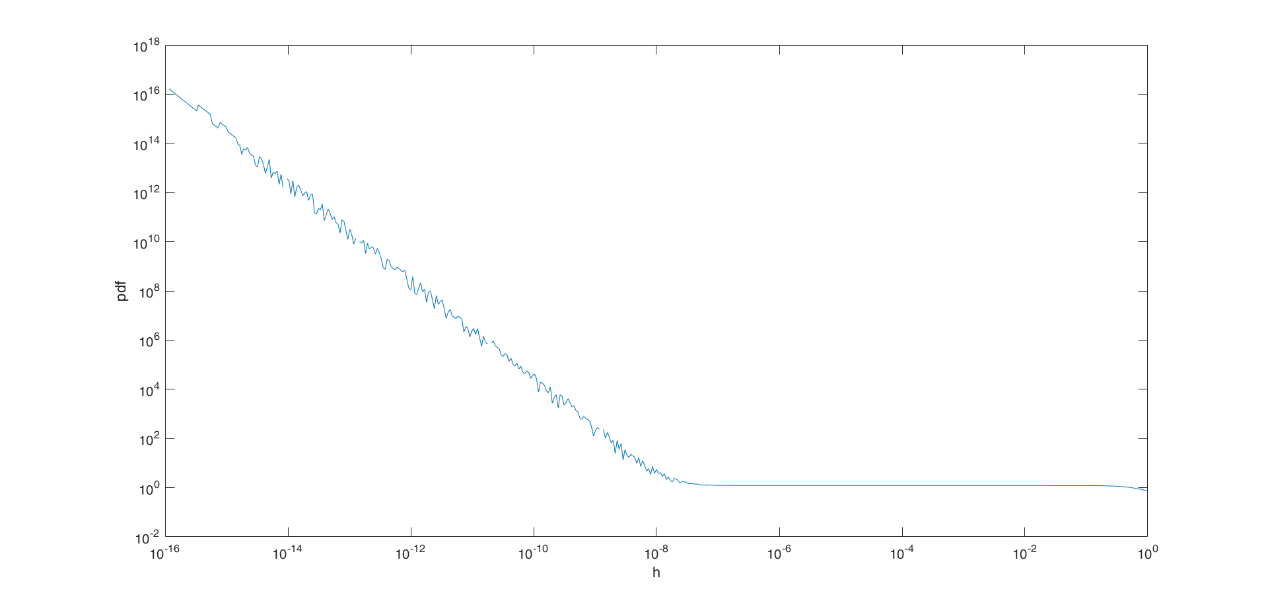
2.7.1 Implementation

Steps of implementation:

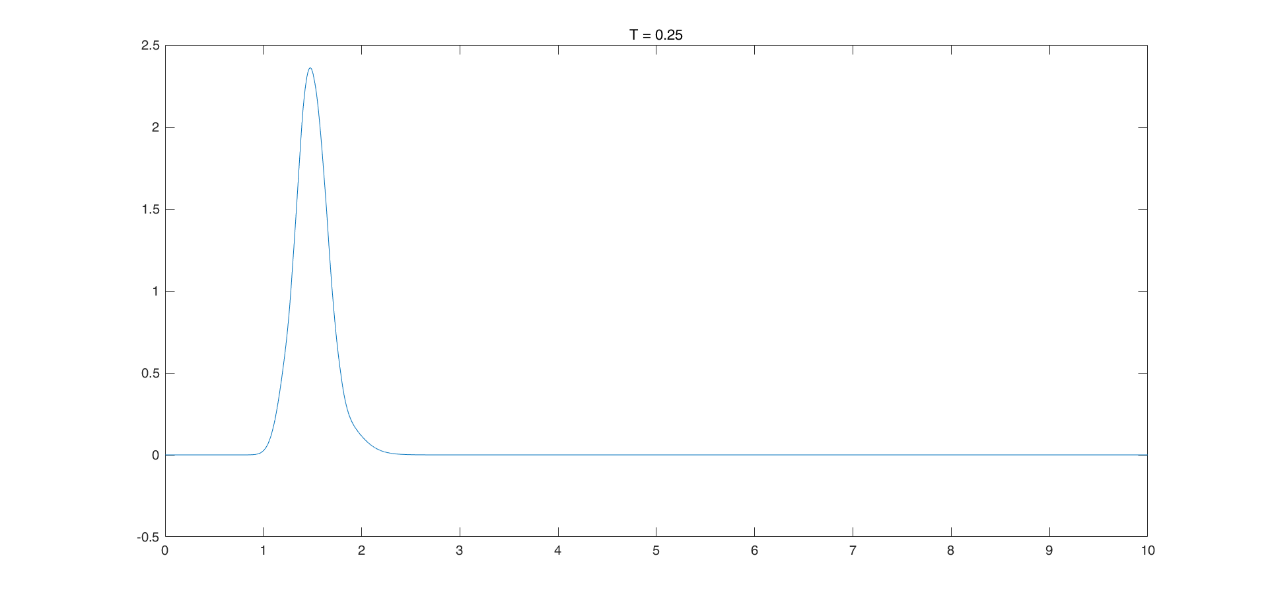
1. Assert validity of function input arguments.
2. Pre-allocate space to the vector of PDF values.
3. Create a matrix of *Ks-h, Ks* and *Ks+h,* where *k* is the length of *Ks*, h is 0.0001.
4. Obtain *Vs* using the function *getVol*, and then obtain the forward prices of the call options *Cs* using the function *getBlackCall*.
5. If , then the PDF value is set to 0. If not, compute the numerical second derivatives using the MATLAB function *diff.*

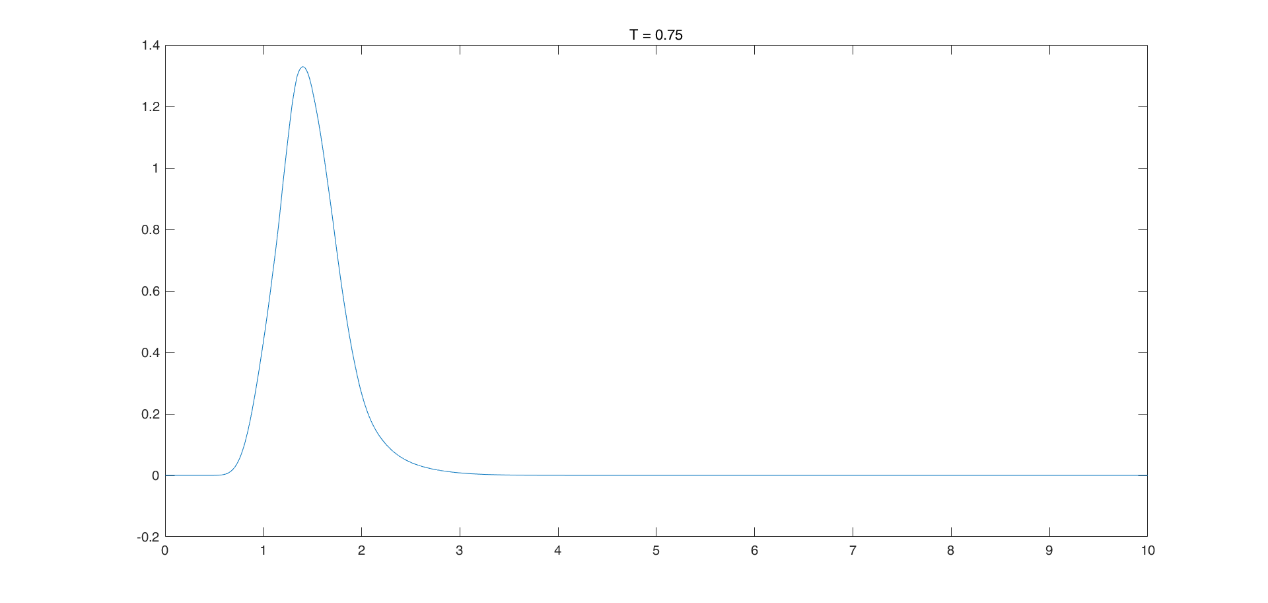
The approximation of the second derivative used in the function is

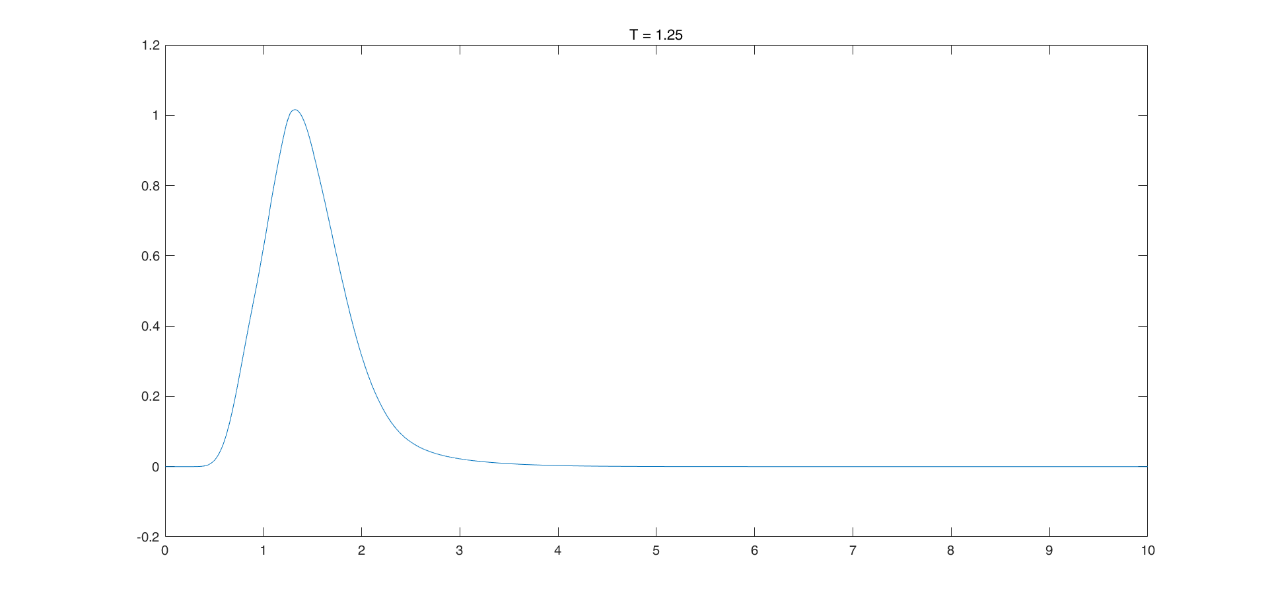
We implement the function with T=0.8, Ks=fwd, using bump size raging from 1e-16 to 1e+0, then plot the PDF value. We can see that the PDF value decreases wiggly at first and becomes stable thereafter. Thus, is set in the function.

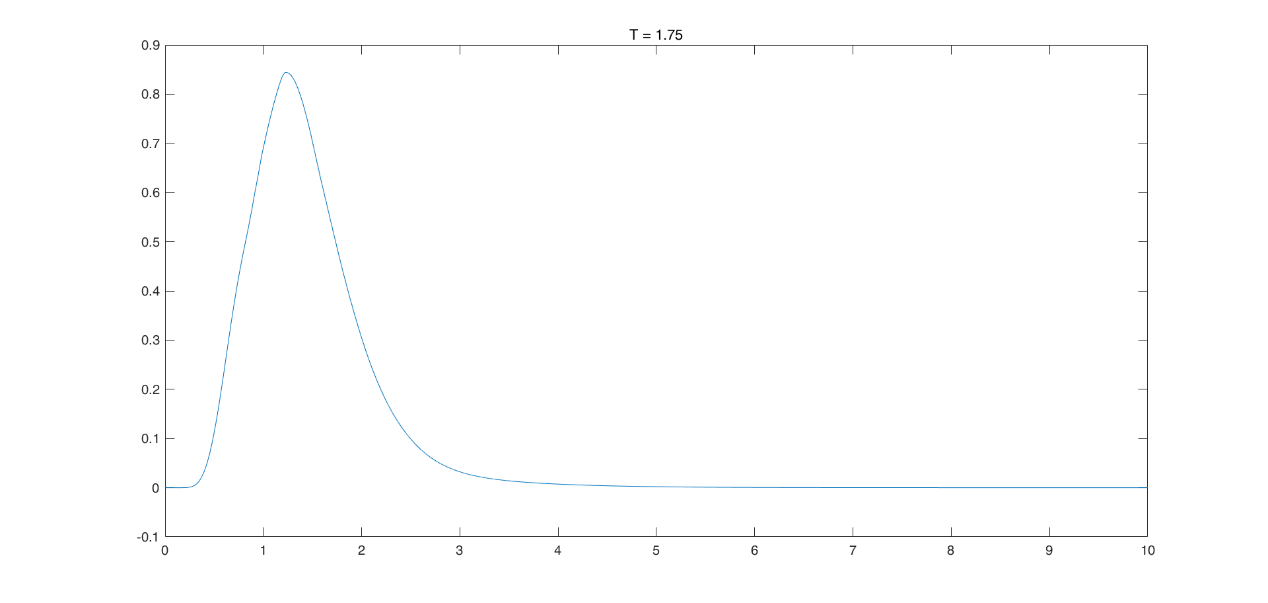


We plot the PDF for each expiry time.









We can see that the PDF for each expiry time behaves like a log-normal distribution as expected. Moreover, for a given volatility surface, the kurtosis of the PDF decreases as the expiry time T increases, which indicates fewer extreme outliers in the distribution.

2.7.2 Tests

We implement the function *testGetPdf* to test properties of the function *getPdf*.

First, we test if the input check works in the function. We design the test case for each input argument, and then implement the function with the invalid inputs. The pass criterion is that a meaningful error message is displayed.

Second, we test if the integral of the PDF is 1 by using the function *getEuropean* with the input argument payoff = @(x) 1.

Third, we compare the mean of the PDF against the forward spot price. We compute the mean by using the function *getEuropean* with the input argument payoff = @(x) x.

The pass criterion for the last two tests stated above is that the absolute difference is not greater than 0.00001.

2.8 Compute forward prices of European options

2.8.1 Implementation

Step 1: Assert validity of function input arguments.

Step 2: We create a handle for the integrand .

Step 3: we use the MATLAB function *integral* to compute numerical integration. If the number of the input arguments is smaller than 4, or the fourth input argument *ints* is [0, +Inf], we compute the integration from 0 to Inf. If a repartition of the integration interval is specified, then we compute the integration in sub-intervals separately, and obtain the sum.

2.8.2 Tests

We implement the function *testGetEuropean* to test properties of the function *getEuropean*.

First, we test if the input check works in the function. We design the test case for each input argument, and then implement the function with the invalid inputs. The pass criterion is that a meaningful error message is displayed.

Second, we compare the forward price of a call option obtained with Black formula against the price obtained by numerical integration. The pass criterion is that the absolute difference is not greater than 0.00001.