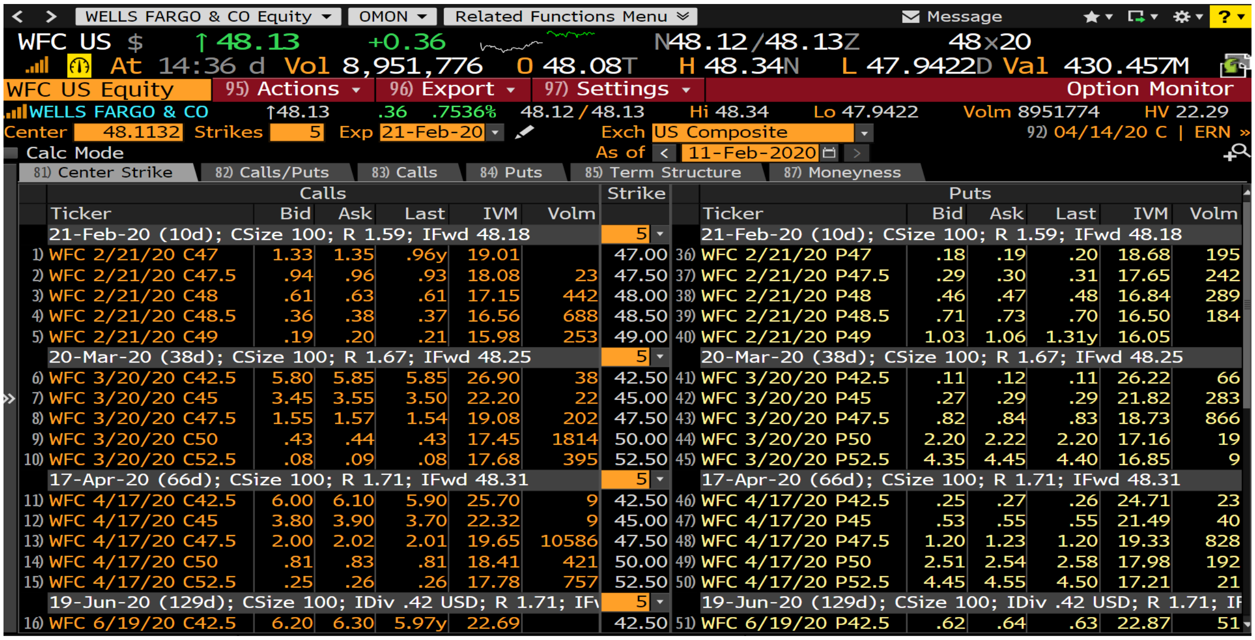
COMPUTER ASSIGNMENT

GOURPRABH, NIKO HIANANTO, SANKET ABHYANKAR, TINGTING ZHOU

1. Bloomberg Screencut

QUESTION 1



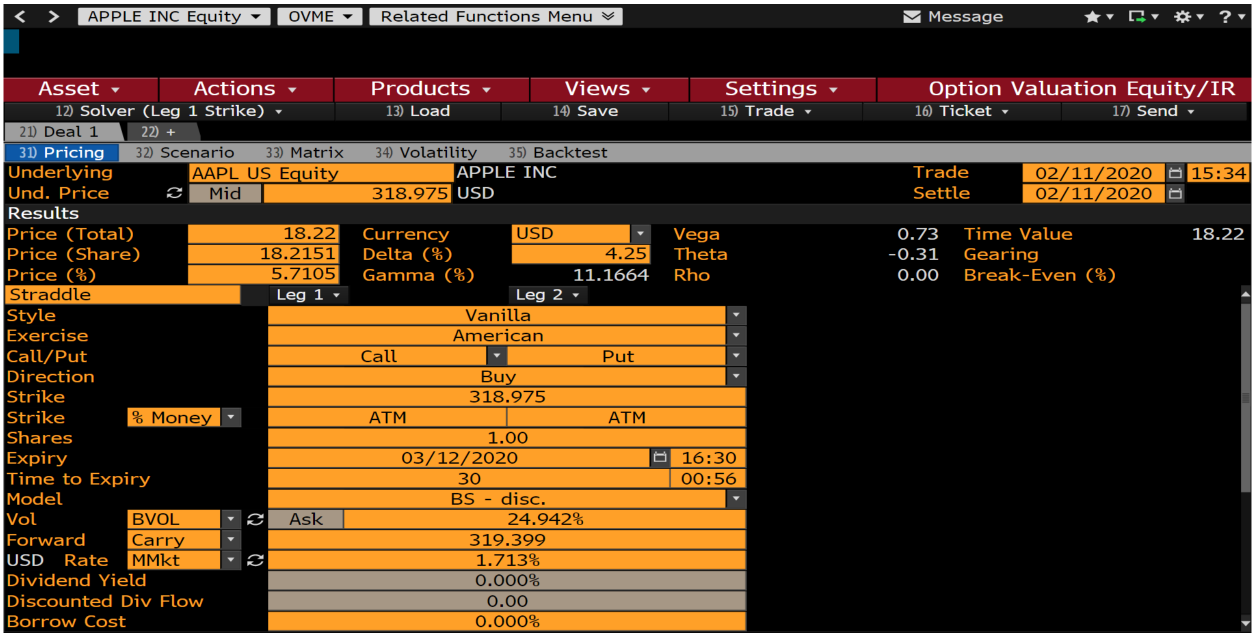
WELLS FARGO & CO Equity | OMON | Related Functions Menu | Message
WFC US $ ↑48.13 +0.36 N48.12/48.13Z 48×20
At 14:36 d Vol 8,951,776 O 48.08T H 48.34N L 47.9422D Val 430.457M
WFC US Equity 95) Actions 96) Export 97) Settings Option Monitor
WELLS FARGO & CO ↑48.13 .36 .7536% 48.12/48.13 Hi 48.34 Lo 47.9422 Volm 8951774 HV 22.29
Center 48.1132 Strikes 5 Exp 21-Feb-20 Exch US Composite 92) 04/14/20 C | ERN »
Calc Mode As of 11-Feb-2020
81) Center Strike 82) Calls/Puts 83) Calls 84) Puts 85) Term Structure 87) Moneyness

	Calls						Strike		Puts					
	Ticker	Bid	Ask	Last	IVM	Volm			Ticker	Bid	Ask	Last	IVM	Volm
	21-Feb-20 (10d); CSize 100; R 1.59; IFwd 48.18						5		21-Feb-20 (10d); CSize 100; R 1.59; IFwd 48.18					
1)	WFC 2/21/20 C47	1.33	1.35	.96y	19.01		47.00	36)	WFC 2/21/20 P47	.18	.19	.20	18.68	195
2)	WFC 2/21/20 C47.5	.94	.96	.93	18.08	23	47.50	37)	WFC 2/21/20 P47.5	.29	.30	.31	17.65	242
3)	WFC 2/21/20 C48	.61	.63	.61	17.15	442	48.00	38)	WFC 2/21/20 P48	.46	.47	.48	16.84	289
4)	WFC 2/21/20 C48.5	.36	.38	.37	16.56	688	48.50	39)	WFC 2/21/20 P48.5	.71	.73	.70	16.50	184
5)	WFC 2/21/20 C49	.19	.20	.21	15.98	253	49.00	40)	WFC 2/21/20 P49	1.03	1.06	1.31y	16.05	
	20-Mar-20 (38d); CSize 100; R 1.67; IFwd 48.25						5		20-Mar-20 (38d); CSize 100; R 1.67; IFwd 48.25					
6)	WFC 3/20/20 C42.5	5.80	5.85	5.85	26.90	38	42.50	41)	WFC 3/20/20 P42.5	.11	.12	.11	26.22	66
7)	WFC 3/20/20 C45	3.45	3.55	3.50	22.20	22	45.00	42)	WFC 3/20/20 P45	.27	.29	.29	21.82	283
8)	WFC 3/20/20 C47.5	1.55	1.57	1.54	19.08	202	47.50	43)	WFC 3/20/20 P47.5	.82	.84	.83	18.73	866
9)	WFC 3/20/20 C50	.43	.44	.43	17.45	1814	50.00	44)	WFC 3/20/20 P50	2.20	2.22	2.20	17.16	19
10)	WFC 3/20/20 C52.5	.08	.09	.08	17.68	395	52.50	45)	WFC 3/20/20 P52.5	4.35	4.45	4.40	16.85	9
	17-Apr-20 (66d); CSize 100; R 1.71; IFwd 48.31						5		17-Apr-20 (66d); CSize 100; R 1.71; IFwd 48.31					
11)	WFC 4/17/20 C42.5	6.00	6.10	5.90	25.70	9	42.50	46)	WFC 4/17/20 P42.5	.25	.27	.26	24.71	23
12)	WFC 4/17/20 C45	3.80	3.90	3.70	22.32	9	45.00	47)	WFC 4/17/20 P45	.53	.55	.55	21.49	40
13)	WFC 4/17/20 C47.5	2.00	2.02	2.01	19.65	10586	47.50	48)	WFC 4/17/20 P47.5	1.20	1.23	1.20	19.33	828
14)	WFC 4/17/20 C50	.81	.83	.81	18.41	421	50.00	49)	WFC 4/17/20 P50	2.51	2.54	2.58	17.98	192
15)	WFC 4/17/20 C52.5	.25	.26	.26	17.78	757	52.50	50)	WFC 4/17/20 P52.5	4.45	4.55	4.50	17.21	21
	19-Jun-20 (129d); CSize 100; IDiv .42 USD; R 1.71; IF						5		19-Jun-20 (129d); CSize 100; IDiv .42 USD; R 1.71; IF					
16)	WFC 6/19/20 C42.5	6.20	6.30	5.97y	22.69		42.50	51)	WFC 6/19/20 P42.5	.62	.64	.63	22.87	51

We can observe that for the same maturity, as the strike price is inceased, the call option price decreases and the put option value increases. For example, for 10 day maturity, as we increase the strike price from 47 to 47.5, we can see that the call ask price decreases from 1.35 to 0.96 while the put ask price increases from 0.19 to 0.30. One explanantion for this is as the strike price is increased , the call option has less payoff while the put option has more payoff.

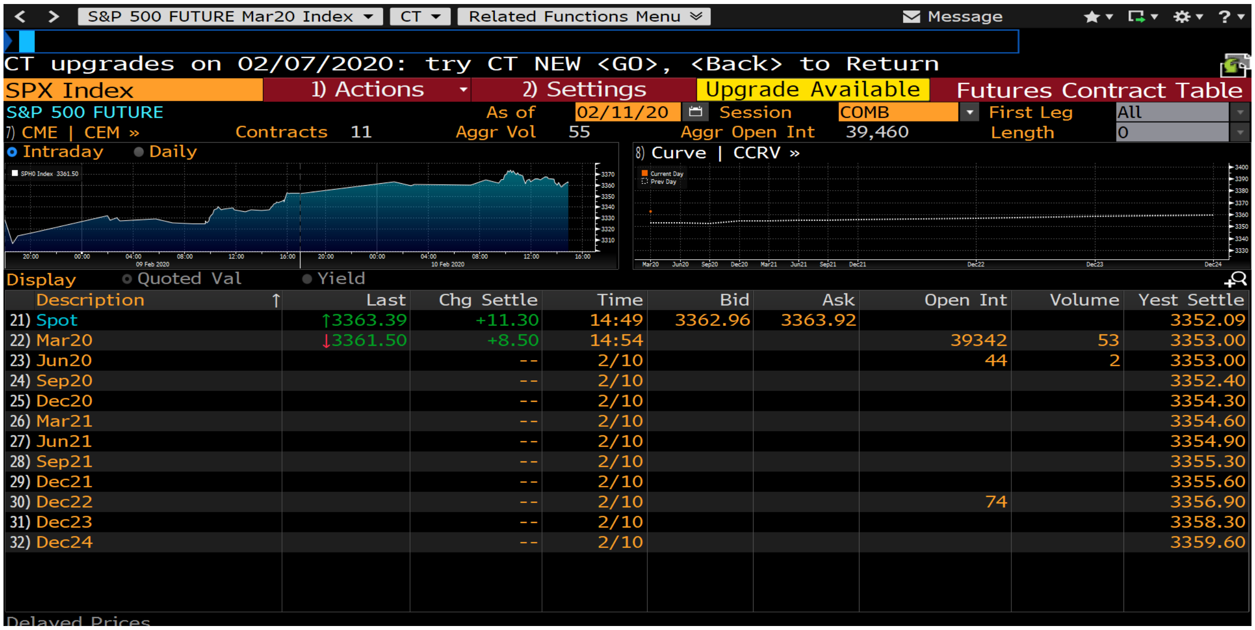
Also, we can observe that for the same exercise price, as the maturity is inceased, both the call option price and the put option value increases. For example, for strike price $47.5, as we increase the maturity from 10d to 38d, we can see that the call ask price increases from 0.96 to 1.57 while the put ask price increases from 0.30 to 0.84. One explanantion for this is as the maturity is increased , the uncertainty (volatility) effect is bigger than the time value effect and that's why the put price increases too.

Question 2



We can see that the vega of the straddle is 0.73 which means for a $1 change in implied volatility, the straddle price will change by $0.73. That means the straddle price is sensitive to the volatility of the underlying asset. and we usually gain profit from a straddle under a market with huge volatility.

Question 3

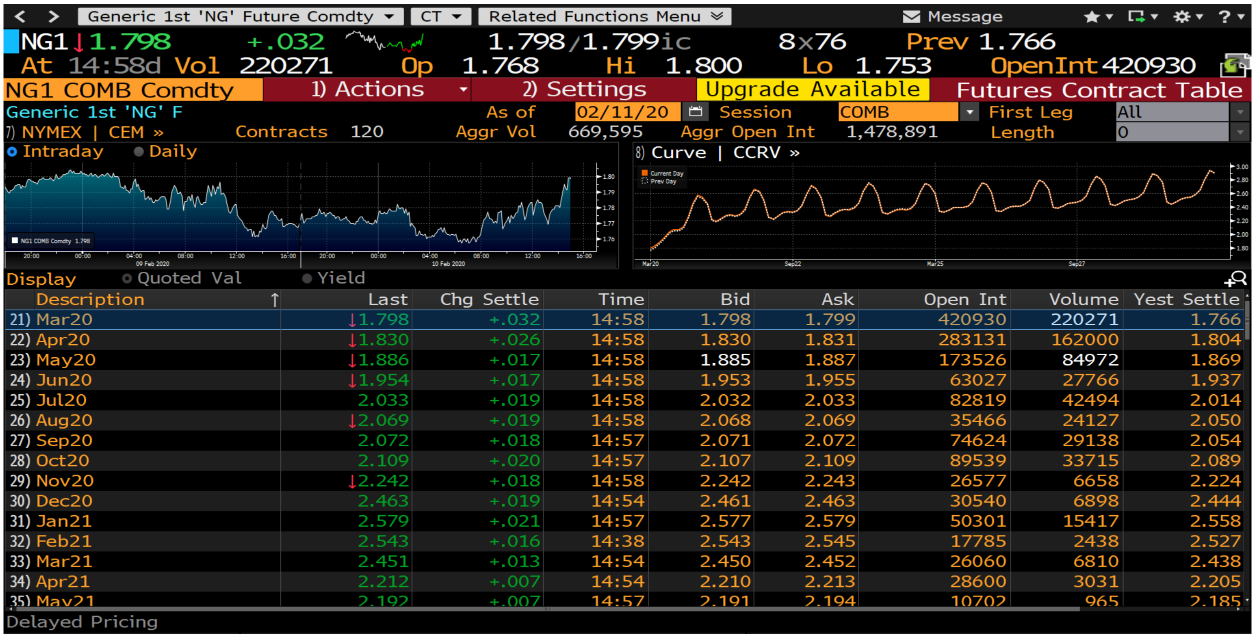


Description	Last	Chg Settle	Time	Bid	Ask	Open Int	Volume	Yest Settle
21) Spot	↑3363.39	+11.30	14:49	3362.96	3363.92			3352.09
22) Mar20	↓3361.50	+8.50	14:54			39342	53	3353.00
23) Jun20		--	2/10			44	2	3353.00
24) Sep20		--	2/10					3352.40
25) Dec20		--	2/10					3354.30
26) Mar21		--	2/10					3354.60
27) Jun21		--	2/10					3354.90
28) Sep21		--	2/10					3355.30
29) Dec21		--	2/10					3355.60
30) Dec22		--	2/10			74		3356.90
31) Dec23		--	2/10					3358.30
32) Dec24		--	2/10					3359.60

We observe that the last price for Mar20 price is 3361, that is, the S&P index is expected to decrease a little bit, but basically the same. Also, the open interest for Mar20 futures is huge, but for Jun20 is pretty small, indicating the liquidity of Mar20 is higher and its demand is huge.

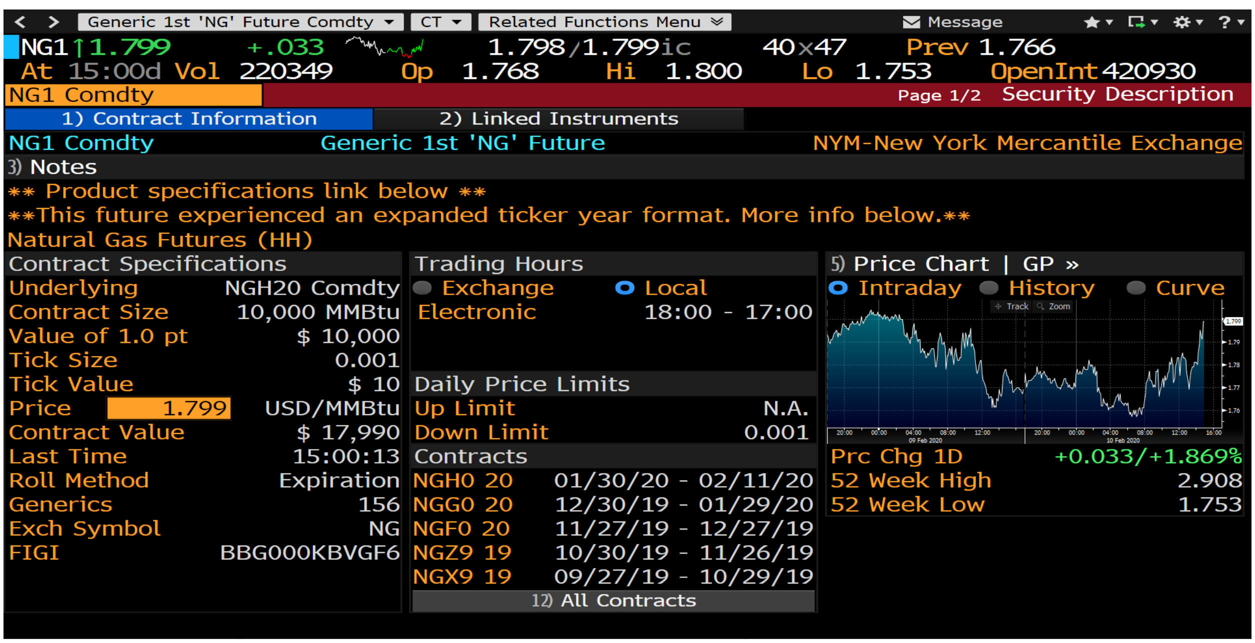
We pay our attention to the margin requirement, and find that the margin requirement for the speculators is higher than hedgers'. Since speculators are more risky investors, the broker tends to ask for a higher margin.

Question 4



Description	Last	Chg Settle	Time	Bid	Ask	Open Int	Volume	Yest Settle
21) Mar20	1.798	+.032	14:58	1.798	1.799	420930	220271	1.766
22) Apr20	1.830	+.026	14:58	1.830	1.831	283131	162000	1.804
23) May20	1.886	+.017	14:58	1.885	1.887	173526	84972	1.869
24) Jun20	1.954	+.017	14:58	1.953	1.955	63027	27766	1.937
25) Jul20	2.033	+.019	14:58	2.032	2.033	82819	42494	2.014
26) Aug20	2.069	+.019	14:58	2.068	2.069	35466	24127	2.050
27) Sep20	2.072	+.018	14:57	2.071	2.072	74624	29138	2.054
28) Oct20	2.109	+.020	14:57	2.107	2.109	89539	33715	2.089
29) Nov20	2.242	+.018	14:58	2.242	2.243	26577	6658	2.224
30) Dec20	2.463	+.019	14:54	2.461	2.463	30540	6898	2.444
31) Jan21	2.579	+.021	14:57	2.577	2.579	50301	15417	2.558
32) Feb21	2.543	+.016	14:38	2.543	2.545	17785	2438	2.527
33) Mar21	2.451	+.013	14:54	2.450	2.452	26060	6810	2.438
34) Apr21	2.212	+.007	14:54	2.210	2.213	28600	3031	2.205
35) May21	2.192	+.007	14:57	2.191	2.194	10702	965	2.185

The natural gas option market appears to be a efficient market, beacuse we can tell that this market has low transation cost ith low bid-ask spread, and since the trading volum is huge, it shows high liquidity.



We observe that there is no upper price limits and the lower limit is just 0.01, also the 52 week highest and lowest shows big difference, indicating the commodity market high volatility.

2. European Options

Question 1.

Straddle with T = 4, r = 0.02, h = 0.25, u = $e^{rh+0.2\sqrt(h)}$, d = $e^{rh-0.2\sqrt(h)}$, S_0 = 100 and K = 90.

```
## [1] 18.48912
```

The composition of stock for each period is shown in the table below

Table 1: Option Delta

1	2	3	4
0.537	0.842	1.000	1.0
	0.200	0.667	1.0
		-0.318	0.3
			-1.0

And the bond composition at each node is given in the table below

Table 2: Option Bond

1	2	3	4
-35.196	-69.27	-89.104	-89.551
	-4.70	-51.984	-89.551
		38.039	-18.489
			89.551

Question 2

Straddle with T = 40, r = 0.02, h = 0.025, u = $e^{rh+0.2\sqrt(h)}$, d = $e^{rh-0.2\sqrt(h)}$, S_0 = 100 and K = 90.

```
## [1] 17.75545
```

The composition of stock for the first 5 periods is shown in the table below

Table 3: Option Delta

1	2	3	4	5
0.536	0.633	0.718	0.791	0.850
	0.437	0.545	0.643	0.729
		0.325	0.444	0.554
			0.202	0.329
				0.070

And the bond composition at each node is given in the table below

Table 4: Option Bond

1	2	3	4	5
-35.878	-45.873	-54.982	-63.009	-69.827
	-26.230	-37.093	-47.259	-56.464
		-15.730	-27.280	-38.387
			-4.556	-16.545
				7.057

Question 3.

Binary call option with T = 4, r = 0.02, h = 0.25, u = $e^{rh+0.2\sqrt(h)}$, d = $e^{rh-0.2\sqrt(h)}$, S_0 = 100 and K = 90.

```
## [1] 0.636274
```

The composition of stock for each period is shown in the table below

Table 5: Option Delta

1	2	3	4
0.019	0.012	0.000	0.000
	0.027	0.026	0.000
		0.028	0.054
			0.000

And the bond composition at each node is given in the table below

Table 6: Option Bond

1	2	3	4
-1.285	-0.513	0.990	0.995
	-1.997	-1.877	0.995
		-2.124	-4.494
			0.000

3. American Tree

American put with T = 250, h = 1/365, r = 0.01, u = $e^{rh+0.15\sqrt(h)}$, d = $e^{rh-0.15\sqrt(h)}$, S_0 = 10 and K = 10.

```
## [1] 0.4653426
```

The composition of stock for the first 5 periods is shown in the table below

Table 7: Option Delta

1	2	3	4	5
-0.46	-0.434	-0.409	-0.384	-0.359
	-0.486	-0.460	-0.434	-0.409
		-0.512	-0.486	-0.460
			-0.538	-0.512
				-0.564

And the bond composition at each node is given in the table below

Table 8: Option Bond

1	2	3	4	5
5.066	4.808	4.550	4.294	4.039
	5.322	5.064	4.805	4.546
		5.579	5.321	5.062
			5.835	5.579
				6.089

American call with T = 250, h = 1/365, r = 0.01, u = $e^{rh+0.15\sqrt(h)}$, d = $e^{rh-0.15\sqrt(h)}$, S_0 = 10 and K = 10.

```
## [1] 0.5285973
```

Table 9: Option Delta

1	2	3	4	5
0.547	0.572	0.596	0.621	0.645
	0.521	0.547	0.572	0.597
		0.496	0.521	0.547
			0.471	0.496
				0.445

The composition of stock for the first 5 periods is shown in the table below

And the bond composition at each node is given in the table below

Table 10: Option Bond

1	2	3	4	5
-4.938	-5.190	-5.442	-5.693	-5.943
	-4.688	-4.940	-5.193	-5.446
		-4.439	-4.690	-4.942
			-4.190	-4.439
				-3.944

4. Discrete Dividends

Question 1.

American put with dividends

```
## [1] 1.455761
```

The composition of stock for the first 5 periods is shown in the table below

Table 11: Option Delta first 5 periods

1	2	3	4	5
-0.705	-0.689	-0.672	-0.654	-0.635
	-0.721	-0.706	-0.690	-0.673
		-0.737	-0.723	-0.708
			-0.751	-0.738
				-0.764

And the bond composition at each node is given in the table below

Table 12: Option Bond first 5 periods

1	2	3	4	5
8.505	8.344	8.170	7.984	7.786
	8.667	8.519	8.357	8.184
		8.817	8.681	8.532
			8.954	8.830
				9.080

American call with dividends

```
## [1] 0.3442288
```

Table 13: Option Delta first 5 periods

1	2	3	4	5
0.513	0.564	0.616	0.667	0.716
	0.460	0.511	0.564	0.616
		0.408	0.458	0.510
			0.357	0.405
				0.309

The composition of stock for the first 5 periods is shown in the table below

And the bond composition at each node is given in the table below

Table 14: Option Bond first 5 periods

1	2	3	4	5
-4.784	-5.303	-5.829	-6.355	-6.870
	-4.266	-4.776	-5.304	-5.841
		-3.756	-4.249	-4.768
			-3.263	-3.730
				-2.797

Question 2.

American straddle with dividends

```
## [1] 1.643528
```

Table 15: American Straddle v. Call + Put

	Price
Straddle	1.64353
Put	1.45576
Call	0.34423
Call + Put	1.79999

As seen on the table above the price of an American Straddle is lower than the price of the American Call & Put combined. This is because when you exercise an American Straddle you have to exercise both at the same time whereas buying an American Call & Put allows you to exercise the different options at different dates.

5. Monte Carlo

The price of the Asian option using the Monte Carlo simulation is given by

```
## [1] 3.33839
```

and the 95% confidence interval (using the t-distribution) of the price is

Table 16: Price Confidence Interval

Lower	Predicted	Upper
3.282221	3.33839	3.394559

6. Pricing American Options using the Longstaff and Schwartz Least-Square Method

Question 1.

American Put Option N = 250, and paths = 100,000

```
## [1] 27.60093
```

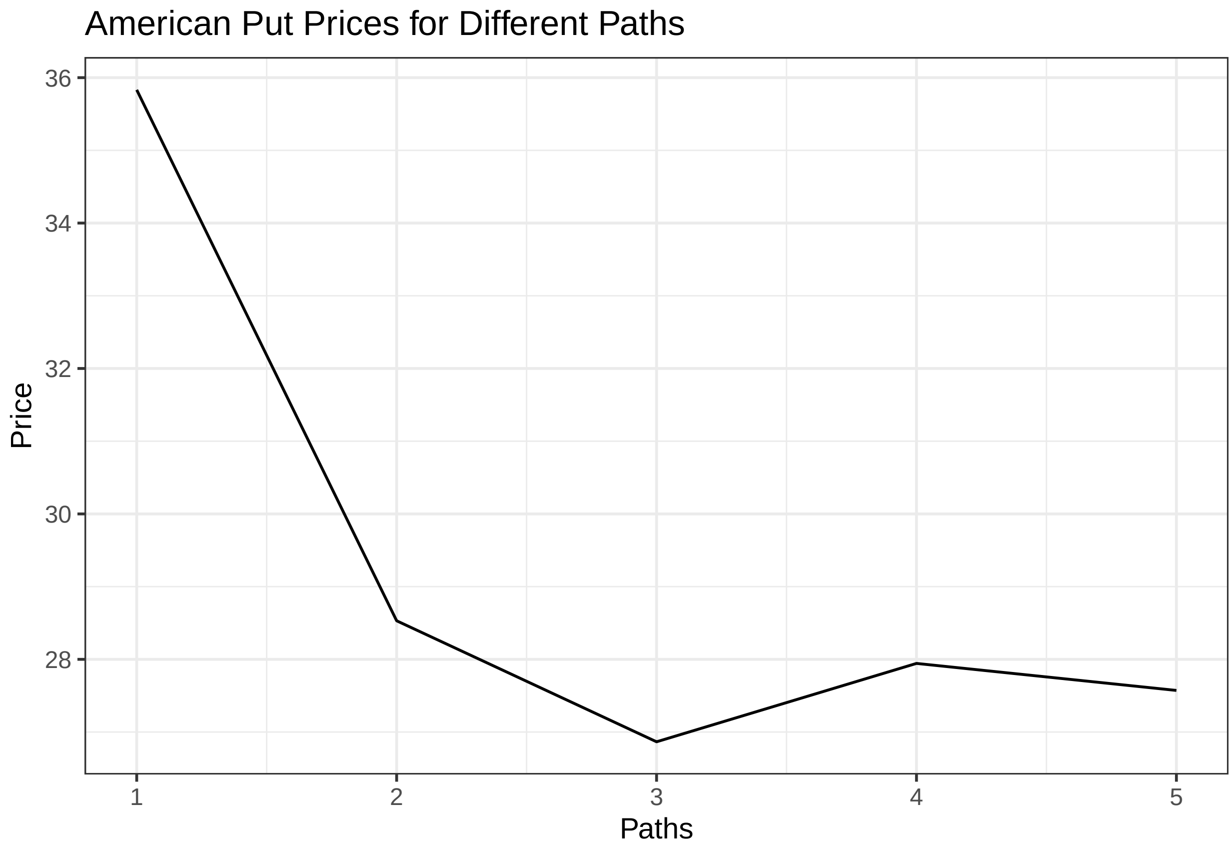
Question 2.

Number of paths of 10, 100, 1,000, 10,000, and 100,000.

Table 17: American Put Prices for Different Path Amounts

Paths	Prices
10	35.83234
100	28.53042
1000	26.86584
10000	27.94384
100000	27.57252

The prices for the American put option for the different paths is shown in the plot below (x is in log scale)

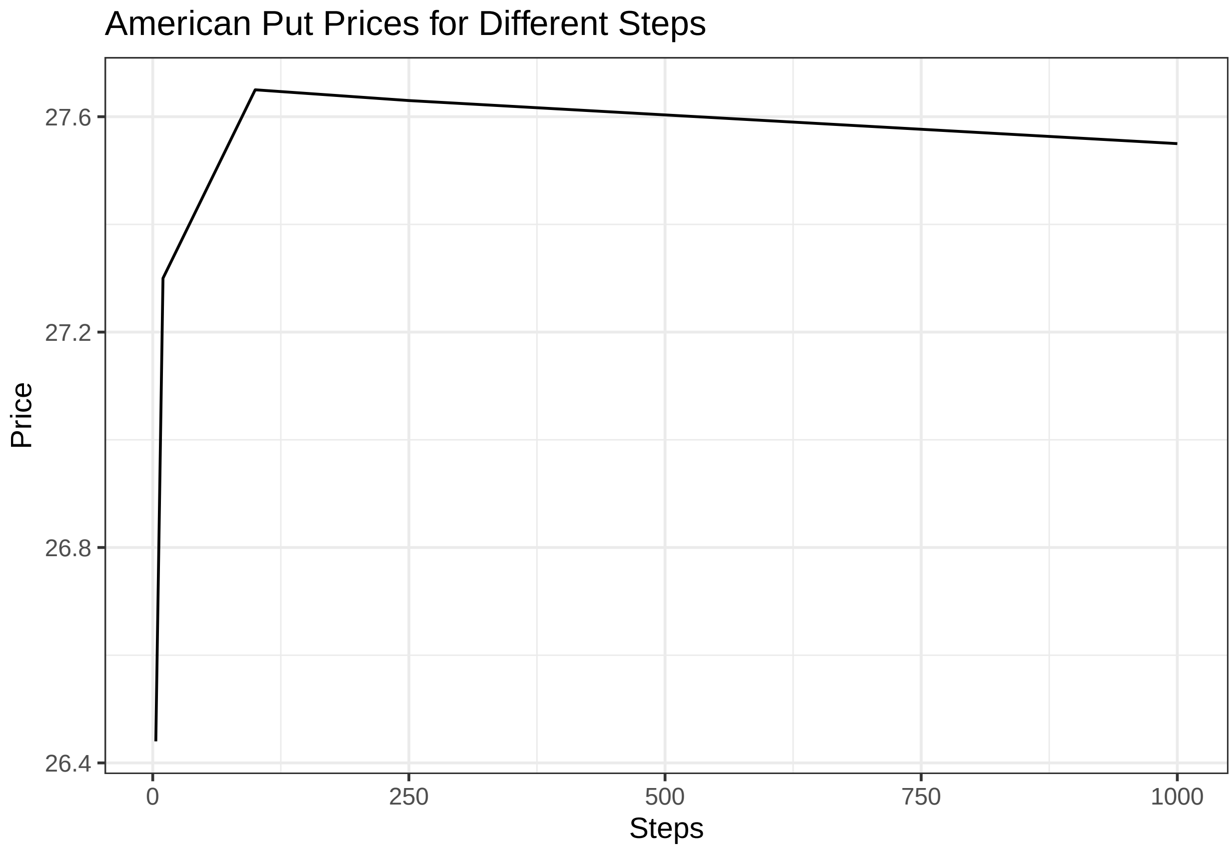


Question 3. Number of paths = 100,000 and steps take values of 3, 10, 100, 250, and 1,000

Table 18: American Put Prices for Steps Amounts

Steps	Prices
3	26.44
10	27.30
100	27.65
250	27.63
1000	27.55

The prices for the American put option for the different paths is shown in the plot below


American Put Prices for Different Steps
27.6
27.2
26.8
26.4
Price
0
250
500
750
1000
Steps

Appendix

```
# tree <- function(s, k, payoff_FUN, r, h, u, d, n, type = 'e'){
#
#   # s = stock price
#   # payoff_FUN = payoff function pass thru
#   # r = interest rate
#   # h = length of each period
#   # u = up factor
#   # d = down factor
#   # n = # of periods
#   # type = option type "a" or "e" american/european respectively
#
#   #create pricing function
#   prc <- function(r, pu, pd, u, d, s, h = h){
#     p = (exp(r*h)-d)/(u-d)
#     prc = exp(-r*h) *(pu *p  + pd * (1-p))
#     delta = (pu-pd)/(s * (u-d))
#     bond = exp(-r*h) *(pd*u - pu*d)/(u-d)
#     return(c(prc, delta, bond))
#   }
#
#   n = n + 1
#   ones = matrix(rep(1, n*n), ncol = n)
#
#   #create up matrix
#   up = ones
#   up[lower.tri(ones, T)] = 0
#   up = t(apply(up, 1, cumsum))
#
#   #down matrix
#   down = diag(c(0,seq(1,n-1)))
#   down = t(apply(down, 1, cumsum))
#
#   #stock path matrix
#   mat = s * u^up * d^down
#   mat[lower.tri(mat, F)] = 0
#
#   #placeholders
#   pl_mat = matrix(rep(0,n*n),ncol =n)
#   opt_mat = delta_mat = bond_mat = matrix(rep(0,(n-1)*(n-1)),ncol = (n-1))
#
#   #set last path of payoff matrix as the payoff of the last stock paths
#   pl_mat[,n] = mat[,n]
#   pl_mat[,n] = payoff_FUN(pl_mat[,n],k)
#
#   #loop through the last columns
#   for(j in (dim(mat)[2]-1):1){ #from last column - 1 : 1
#
#     #loop through the rows (1 to j); no. rows decreasing every period by 1 (recombining tree)
#     for(i in 1 : j){
#       #for each row calculate the price, delta and bond using the prc function
#       #pu is the up payoff defined as the next column and the same row
```

```
#       #pd is the down payoff defined as the next column and the next row
#       #mat[i,j] is the current stock price
#
#       val = prc(r, pu = pl_mat[i,j+1], pd = pl_mat[i+1,j+1], u, d, s = mat[i,j], h)
#
#       if(tolower(type) == 'a'){ #american option
#
#         #max of exercise that date or value
#         pl_mat[i,j] = max(payoff_FUN(mat[i,j], k) , val[1])
#         if(payoff_FUN(mat[i,j], k) > val[1]) opt_mat[i,j] = 1
#
#       } else if(tolower(type) == 'e'){#european option
#         #simply the intrinsic value
#         pl_mat[i,j] = val[1]
#       } else{
#         stop('choose correct option type')
#       }
#
#
#       delta_mat[i,j] = val[2]
#       bond_mat[i,j] = val[3]
#
#
#     }
#   }
#
#   return(list("stock" = mat, "payoff" = pl_mat, "delta" = delta_mat, "bond" = bond_mat, "optimal" = op
# }
#
# #create common option payoff functions (vectorized)
# straddle_payoff <- function(s, k){
#   sapply(s, function(x) max(x-k,k - x))
# }
#
# binary_call <- function(s, k){
#   sapply(s, function(x) ifelse(x > k, 1, 0))
# }
#
#
# call_payoff <- function(s,k){
#   sapply(s, function(x) max(x-k,0))
# }
#
# put_payoff <- function(s, k){
#   sapply(s, function(x) max(k-x,0))
# }
#
# # problem 2
# ## question 1
# n = 4
# r = 0.02
# h = 0.25
# u = exp(r * h + 0.2 *sqrt(h))
```

```r
# d = exp(r * h - 0.2 *sqrt(h))
# s = 100
# k = 90
#
# straddle_1 = tree(s = s, k = k, straddle_payoff, r = r, h = h, u = u, d = d, n = n, type = 'e')
# straddle_1$payoff[1,1]
# # stock
# delta_print = straddle_1$delta
# delta_print[lower.tri(delta_print, F)] = NA
# delta_print %>%
#   output_table(seq(dim(straddle_1$delta)[2]), digits = 3, caption = "Option Delta", align = rep("c",
#
# # bond
# bond_print = straddle_1$bond
# bond_print[lower.tri(bond_print, F)] = NA
# bond_print %>%
#   output_table(seq(dim(straddle_1$bond)[2]), digits = 3, caption = "Option Bond", align = rep("c", di
#
# ## question 2
# n = 40
# r = 0.02
# h = 0.025
# u = exp(r * h + 0.2 *sqrt(h))
# d = exp(r * h - 0.2 *sqrt(h))
# s = 100
# k = 90
#
# straddle_2 = tree(s = s, k = k, straddle_payoff, r = r, h = h, u = u, d = d, n = n, type = 'e')
# straddle_2$payoff[1,1]
# # stock subset first 5
# delta_print = straddle_2$delta[1:5,1:5]
# delta_print[lower.tri(delta_print, F)] = NA
# delta_print %>%
#   output_table(seq(dim(delta_print)[2]), digits = 3, caption = "Option Delta", align = rep("c", dim(d
# # stock subset first 5
# bond_print = straddle_2$bond[1:5, 1:5]
# bond_print[lower.tri(bond_print, F)] = NA
# bond_print %>%
#   output_table(seq(dim(bond_print)[2]), digits = 3, caption = "Option Bond", align = rep("c", dim(bon
#
# # question 3
# n = 4
# r = 0.02
# h = 0.25
# u = exp(r * h + 0.2 *sqrt(h))
# d = exp(r * h - 0.2 *sqrt(h))
# s = 100
# k = 90
#
# binary_call = tree(s = s, k = k, binary_call, r = r, h = h, u = u, d = d, n = n, type = 'e')
# binary_call$payoff[1,1]
# ### stock
# delta_print = binary_call$delta
```

```
# delta_print[lower.tri(delta_print, F)] = NA
# delta_print %>%
#   output_table(seq(dim(binary_call$delta)[2]), digits = 3, caption = "Option Delta", align = rep("c",
# ### bond
# bond_print = binary_call$bond
# bond_print[lower.tri(bond_print, F)] = NA
# bond_print %>%
#   output_table(seq(dim(binary_call$bond)[2]), digits = 3, caption = "Option Bond", align = rep("c", d
#
#
# # problem 3 american options
# ## american put
# h = 1/365
# n  = 250
# r = 0.01
# u = exp(r * h + 0.15 *sqrt(h))
# d = exp(r * h - 0.15 *sqrt(h))
# s = 10
# k = 10
#
# american_put = tree(s = s, k = k, put_payoff, r = r, h = h, u = u, d = d, n = n, type = 'a')
# american_put$payoff[1,1]
# # stock subset first 5
# delta_print = american_put$delta[1:5,1:5]
# delta_print[lower.tri(delta_print, F)] = NA
# delta_print %>%
#   output_table(seq(dim(delta_print)[2]), digits = 3, caption = "Option Delta", align = rep("c", dim(d
# #bond subset first 5
# bond_print = american_put$bond[1:5, 1:5]
# bond_print[lower.tri(bond_print, F)] = NA
# bond_print %>%
#   output_table(seq(dim(bond_print)[2]), digits = 3, caption = "Option Bond", align = rep("c", dim(bon
#
# ## american call
# n = 250
# h = 1/365
# r = 0.01
# u = exp(r * h + 0.15 *sqrt(h))
# d = exp(r * h - 0.15 *sqrt(h))
# s = 10
# k = 10
#
# american_call = tree(s = s, k = k, call_payoff, r = r, h = h, u = u, d = d, n = n, type = 'a')
# american_call$payoff[1,1]
# # stock
# delta_print = american_call$delta[1:5,1:5]
# delta_print[lower.tri(delta_print, F)] = NA
# delta_print %>%
#   output_table(seq(dim(delta_print)[2]), digits = 3, caption = "Option Delta", align = rep("c", dim(d
#
# # bond subset first 5
# bond_print = american_call$bond[1:5, 1:5]
# bond_print[lower.tri(bond_print, F)] = NA
```

```
# bond_print %>%
#   output_table(seq(dim(bond_print)[2]), digits = 3, caption = "Option Bond", align = rep("c", dim(bon
#
# # problem 4
#
# tree_div <- function(s, k, payoff_FUN, r, h, u, d, n, div_yield, div_dt, type = 'e'){
#
#   # s = stock price
#   # k = exercise price
#   # payoff_FUN = payoff function pass thru
#   # r = interest rate
#   # h = length of each period
#   # u = up factor
#   # d = down factor
#   # n = number of periods
#   # type = option type "a" or "e" american/european respectively
#   # div_yield = discrete dividend yield
#   # div_dates = discrete dividend distribution dates
#
#   prc <- function(r, pu, pd, u, d, s, h = h){
#     p = (exp(r*h)-d)/(u-d)
#     prc = exp(-r*h) *(pu *p  + pd * (1-p))
#     delta = (pu-pd)/(s * (u-d))
#     bond = exp(-r*h) *(pd*u - pu*d)/(u-d)
#     return(c(prc, delta, bond))
#   }
#   #initialize stock paths matrix
#   n = n + 1
#   ones = matrix(rep(1, n*n), ncol = n)
#
#   #create up matrix
#   up = ones
#   up[lower.tri(ones, T)] = 0
#   up = t(apply(up, 1, cumsum))
#
#   #create down matrix
#   down = diag(c(0,seq(1,n-1)))
#   down = t(apply(down, 1, cumsum))
#
#   #create dividend matrix
#   div_dt = div_dt + 1 #add by 1 for indexing
#   div_mat = matrix(rep(0, n*n), ncol = n)
#   div_mat[,div_dt] = 1
#   div_mat = t(apply(div_mat, 1, cumsum))
#   div_mat[lower.tri(div_mat, F)] = 0
#
#   #create stock paths (with dividends)
#   mat = s * u^up * d^down * (1 - div_yield)^div_mat
#   mat[lower.tri(mat, F)] = 0
#
#   #placeholders
#   pl_mat = matrix(rep(0,n*n),ncol =n)
#   opt_mat = delta_mat = bond_mat = matrix(rep(0,(n-1)*(n-1)),ncol = (n-1))
```

```
#
#   #end path prices and payoffs
#   pl_mat[,n] = mat[,n]
#   pl_mat[,n] = payoff_FUN(pl_mat[,n],k)
#
#   #loop from end of column
#   for(j in (dim(mat)[2]-1):1){
#     #loop through each row
#
#     for(i in 1 : j){
#
#       val = prc(r, pu = pl_mat[i,j+1], pd = pl_mat[i+1,j+1], u, d, s = mat[i,j], h)
#
#       if(tolower(type) == 'a'){
#
#         pl_mat[i,j] = max(payoff_FUN(mat[i,j], k) , val[1]) #max of exercise that date or value
#         if(payoff_FUN(mat[i,j], k) > val[1]) opt_mat[i,j] = 1
#
#       } else if(tolower(type) == 'e'){
#         pl_mat[i,j] = val[1]
#       } else{
#         stop('choose correct option type')
#       }
#       delta_mat[i,j] = val[2]
#       bond_mat[i,j] = val[3]
#
#
#     }
#   }
#
#   return(list("stock" = mat, "payoff" = pl_mat, "delta" = delta_mat, "bond" = bond_mat, "optimal" = o
# }
# ## american put with discrete dividend
# k = 10
# r = 0.02
# s = 10
# h = 1/365
# u = exp(0.2 * sqrt(h))
# d = 1/u
# n = 200
# div_dt = c(50, 100, 150)
# delta = 0.05
#
# american_put_div = tree_div(s = s, k = k, put_payoff, r = r, h = h, u = u, d = d, n = n, div_yield =
# american_put_div$payoff[1,1]
#
# ## american put with dividend
# k = 10
# r = 0.02
# s = 10
# h = 1/365
# u = exp(0.2 * sqrt(h))
# d = 1/u
```

```
# n = 200
# div_dt = c(50,100,150)
# delta = 0.05
#
# american_call_div = tree_div(s = s, k = k, call_payoff, r = r, h = h, u = u, d = d, n = n, div_yield =
# american_call_div$payoff[1,1]
# ## straddle
# k = 10
# r = 0.02
# s = 10
# h = 1/365
# u = exp(0.2 * sqrt(h))
# d = 1/u
# n = 200
# div_dt = c(50, 100, 150)
# delta = 0.05
#
# american_straddle_div = tree_div(s = s, k = k, straddle_payoff, r = r, h = h, u = u, d = d, n = n, di
# american_straddle_div$payoff[1,1]
#
# # problem 5
# t = 1
# k = 220
# sigma = 0.2
# n = 365
# h = t/n
# s0 = 200
# r = 0.02
# paths = 100000
# sim = matrix(0, ncol = n+1, nrow = paths)
# sim[,1] = s0
# for(i in 2:dim(sim)[2]){
#   #assumge GBM stock price - path follows S_0 * exp((r - sigma^2)h + sigma * sqrt(h) * Z)
#   sim[,i] = sim[,i-1] * exp((r - 0.5 * sigma^2)*h  + rnorm(dim(sim)[1],0, sigma) * sqrt(h))
# }
# # price
# res = apply(sim, 1, function(x) max(mean(x) - k, 0 ))
# y = mean(res)
# y
#
# # 95% confidence level
# err <- qt(0.975,paths -2) * sd(res)/sqrt(paths)
# conf = c(y - err, y, y + err)
#
# matrix(conf, nrow = 1) %>%
#   output_table(colnames = c("Lower","Predicted","Upper"), caption = "Price Confidence Interval",  ali
#
# # problem 6
# sim_path <- function(s, sigma, steps, path_amt, t){
#   paths = path_amt
#   sim_stock_path = matrix(0, ncol = steps+1, nrow = path_amt)
#   sim_stock_path[,1] = s
#   h = t/steps
```

```
#   for(i in 2:dim(sim_stock_path)[2]){
#     sim_stock_path[,i] = sim_stock_path[,i-1] * exp((r - 0.5 * sigma^2)*h  + rnorm(dim(sim_stock_path
#   }
#   return(sim_stock_path)
# }
#
# regress <- function(s, k, payoff_FUN, r, sigma, t, steps, path_amt){
#
#   sim_stock_path = sim_path(s = s, sigma = sigma, steps = steps, path_amt = path_amt, t = t)
#   # sim_stock_path = test
#
#   #cf matrix
#   cf_mat = matrix(0, ncol = steps, nrow = path_amt)
#
#   #helper params
#   dim = dim(cf_mat)
#   h = t/steps
#   disc_rate = exp(-r * h)
#
#   #initialize last cash flow matrix values
#   cf_mat[,dim[2]] = put_payoff(sim_stock_path[,ncol(sim_stock_path)], k)
#
#   #set discount vectors
#   disc_vec = disc_rate ^ (1:steps)
#
#   #loop through and run regression
#   for(i in (steps-1): 1){ #col 1 = initial stock value
#     #current stock price
#     X = sim_stock_path[,i+1] #sim stock path matrix has s_0, add 1 to col index
#     #value if exercise
#     ex = put_payoff(X, k)
#     #itm indicator
#     itm = ifelse(ex > 0, 1, NA)
#
#     #value if not exercised
#     pl = cf_mat[,(i+1):dim[2], drop = F] #take future cf matrix
#     NO = t(t(pl) * disc_rate^(1:dim(pl)[2])) #discount future cf's for each path with the disc rate r
#     NO = apply(NO, 1, sum) * itm #if not in the money set as NA and compress the matrix to a single v
#
#     # run regress
#     # out = lm(NO ~ X + I(X^2), na.action = na.omit)
#     #expected value
#     # expected_val = predict(out, newdata = data.frame(X)) * coalesce(itm, 0)
#
#     # only regress on paths that are in the money at that time
#     itm_NO = NO[!is.na(itm)]
#     if(length(itm_NO) > 0){
#       #run regress
#       out = lm(NO ~ X + I(X^2), na.action = na.omit)
#       #expected value
#       expected_val = predict(out, newdata = data.frame(X)) * coalesce(itm, 0) #set other expected val
#     } else{#if no paths are itm
#       warning(c("No paths are ITM at step",i))
```

```
#       expected_val = 0
#     }
#
#
#     #update cf matrix and optimal matrix if current exercise value > expected value from regression
#     cf_mat[, i] = ex * (ex > expected_val) #set current cf as exercise value if ex > expected value
#     cf_mat[c(ex>expected_val), (i+1):dim[2]] = 0 #set future cf as 0 if current exercise value > expe
#
#   }
#
#
#   prc = mean(apply(t(t(cf_mat) * disc_vec), 1, sum))
#
#   return(prc)
# }
# ## question 1
# s = 200
# r = 0.1
# sigma = 0.3
# k = 220
# t = 1
# n = 250
# paths = 100000
#
# regress(s = s, k = k, put_payoff, r = r, sigma = sigma, steps = n, path_amt = paths, t = t)
#
# ## question 2
# s = 200
# r = 0.1
# sigma = 0.3
# k = 220
# t = 1
# n = 250
# paths = c(10,100,1000,10000,100000)
#
# path_var = sapply(paths, function(x) regress(s = s, k = k, put_payoff, r = r, sigma = sigma, steps = 
#
# ## question 3
# s = 200
# r = 0.1
# sigma = 0.3
# k = 220
# t = 1
# n = c(3,10,100,250,1000)
# paths = 100000
#
# steps_var = sapply(n, function(x) regress(s = s, k = k, put_payoff, r = r, sigma = sigma, steps = x, p
```