```
In[1]:= s = 0.006529828886969226
Out[1]= 0.00652983

In[2]:= edistNorm = NormalDistribution[-0.00011849766017034486`, 0.004097422554612836`]
Out[2]= NormalDistribution[-0.000118498, 0.00409742]

In[3]:= (* 10 m distribution for USDC-ETH *)

In[4]:= edistWethUsdc90dFiltered10MinCandle = StableDistribution[1, 1.4646547203677354`,
          -0.04962071457296463`, -0.000010553047315527044`, 0.0015427669619594313`]
Out[4]= StableDistribution[1, 1.46465, -0.0496207, -0.000010553, 0.00154277]

In[5]:= edistWethUsdc90dFiltered1hCandle = StableDistribution[1,
          1.4646547203677354`, -0.04962071457296463`, -0.000010553047315527044` * 6,
          0.0015427669619594313` * (6 / 1.4646547203677354`) ^ (1 / 1.4646547203677354`)]
Out[5]= StableDistribution[1, 1.46465, -0.0496207, -0.0000633183, 0.00404045]

In[6]:= (* Great *)
        (* Now be super rigorous w it! *)

In[7]:= CDF[StableDistribution[1, 1.46465, -0.0496207, -0.000010553, 0.00154277], 3]
Out[7]= 0.999997

In[8]:= InverseCDF[edistWethUsdc90dFiltered10MinCandle, 0.99]
Out[8]= 0.0124991

In[9]:= mu = -0.00011849766017034486`
Out[9]= -0.000118498

In[10]:= sig = 0.004097422554612836`
Out[10]= 0.00409742

In[11]:= CDF[NormalDistribution[0, 1], 0]
Out[11]= 1/2

In[12]:= CDF[NormalDistribution[0, 1], -sig]
Out[12]= 0.498365

In[13]:= (1 - CDF[NormalDistribution[0, 1], -sig]) / (1 - CDF[NormalDistribution[0, 1], 0])
Out[13]= 1.00327

In[14]:= Log[1.003269261047541`]
Out[14]= 0.00326393
```

```
In[15]:= 0.0032639286325196527` + sig^2 / 2.0
Out[15]= 0.00327232

In[16]:= (* Perfect .... this is > 0 so can use l * Q to make this go to 0 *)

In[17]:= mu
Out[17]= -0.000118498

In[18]:= (* Bring in mu *)

In[19]:= CDF[NormalDistribution[0, 1], -(mu - 2 * s) / sig - sig]
Out[19]= 0.999341

In[20]:= 1 - CDF[NormalDistribution[mu - 2 * s, sig], 0]
Out[20]= 0.000649488

In[21]:= CDF[NormalDistribution[mu - 2 * s, sig], 0]
Out[21]= 0.999351

In[22]:= CDF[NormalDistribution[0, 1], -(mu - 2 * s) / sig]
Out[22]= 0.999351

In[23]:= (1 - CDF[NormalDistribution[0, 1], -(mu - 2 * s) / sig - sig]) /
          (1 - CDF[NormalDistribution[0, 1], -(mu - 2 * s) / sig])
Out[23]= 1.01437

In[24]:= (* even better! Beautiful :) ... mu + sig^2/2 - l * Q + Log[(1-CDF)/(1-CDF)] ≤
          0 to have negative EV at Q *)

In[25]:= Exp[mu + sig^2 / 2.0]
Out[25]= 0.99989

In[26]:= mu + sig^2 / 2.0
Out[26]= -0.000110103

In[27]:= 2 * s
Out[27]= 0.0130597

In[28]:= rhs = mu - 2 * s + sig^2 / 2.0 +
            Log[(1 - CDF[NormalDistribution[0, 1], -(mu - 2 * s) / sig - sig]) /
              (1 - CDF[NormalDistribution[0, 1], -(mu - 2 * s) / sig])]
Out[28]= 0.00110089

In[29]:= (* What's the expected PnL given > s when lambda = 0? *)
```

In[30]:=
```
Exp[rhs] - 1
```
Out[30]= 0.0011015

In[31]:=
```
rhs
```
Out[31]= 0.00110089

In[32]:=
```
(* 0.11% in 10m interval as scalp is pretty small already. Now,
ensure we make that EV ≤ 0 ... *)
```

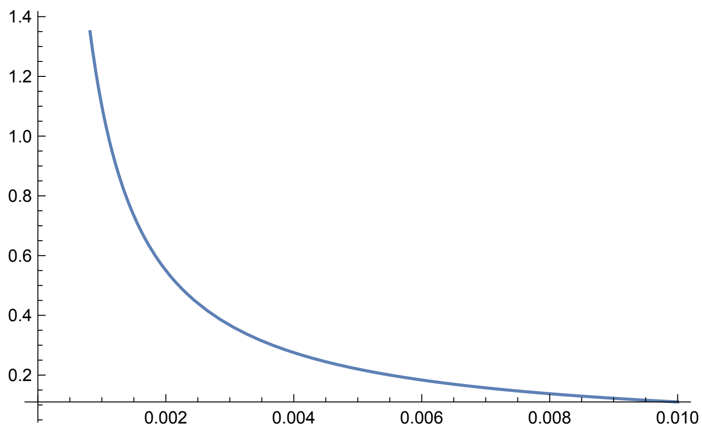
In[33]:=
```
(* need l*Q ≥ rhs for negative EV. Take l'*(Q/Qmax) ≥ rhs ... l' ≥
 rhs * (Qmax/Q) *)
```

In[34]:=
```
(* if take l' = rhs * (Qmax/Q0), know that EV < 0 whenever Q > Q0 *)
```

In[35]:=
```
lNormUsdcWethPrime[q_] := (1 / q) * rhs
```

In[36]:=
```
Plot[lNormUsdcWethPrime[q], {q, 0, 0.01}]
```

Out[36]=



In[37]:=
```
Exp[1.5]
```
Out[37]= 4.48169

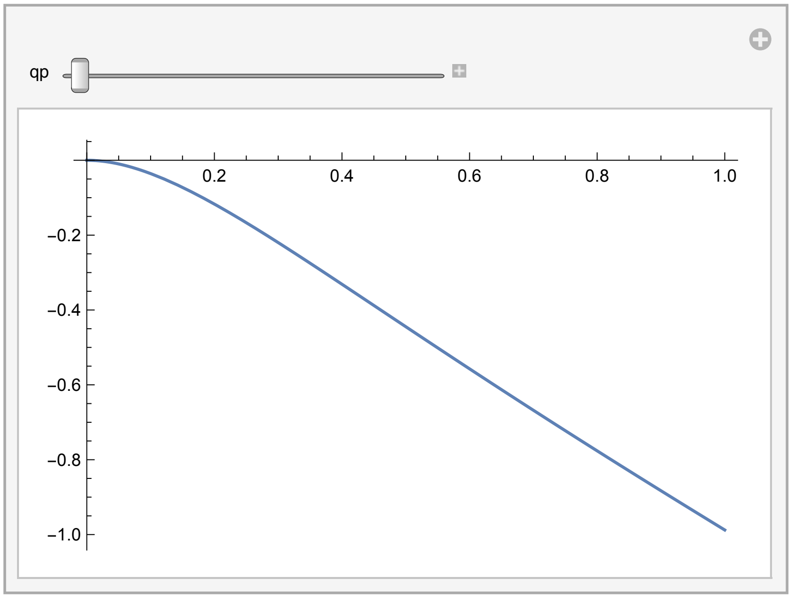
In[38]:=
```
Exp[-1.5]
```
Out[38]= 0.22313

In[39]:=
```
(* Check norm EV values given Q0 choice *)
```

In[40]:=
```
evNormUsdcWeth[qp_, q_] := q * (Exp[rhs - lNormUsdcWethPrime[qp] * q] - 1)
```

In[41]:=
```
Manipulate[Plot[evNormUsdcWeth[qp, q], {q, 0, 1}], {qp, 0.00025, 0.01}]
```

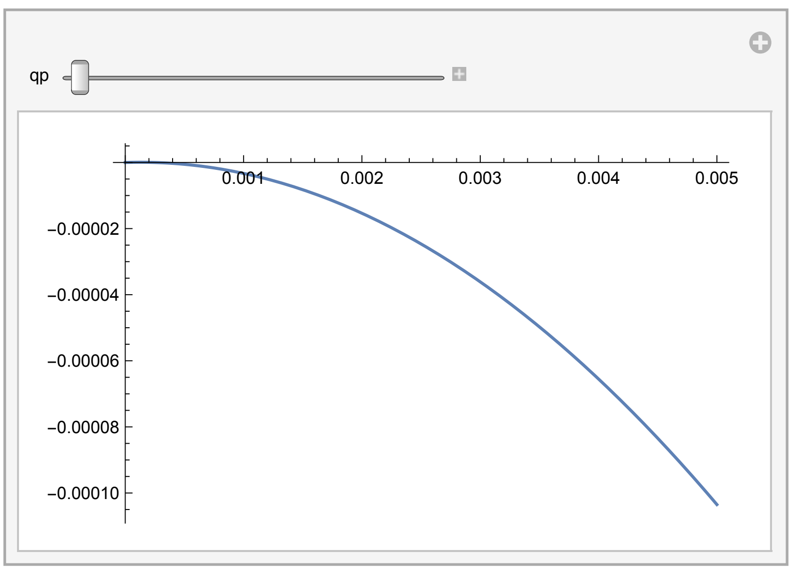
Out[41]=



In[42]:=
```
(* Zoom in around 1% mark ... *)
```

In[43]:=
```
Manipulate[Plot[evNormUsdcWeth[qp, q], {q, 0, 0.005}], {qp, 0.00025, 0.01}]
```

Out[43]=



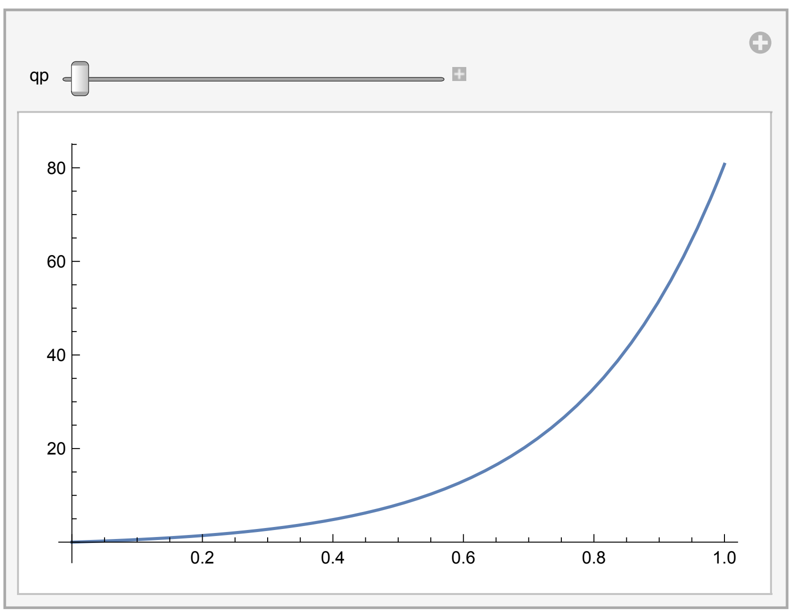
In[44]:=
```
(* EV of < 0.01 bps of OI max/cap on scalp if take lprime to happen at Q =
 1% of cap *)
```

In[45]:=
```
(* Take 0.1% of cap as the l anchor *)
```

In[46]:=
```
lNormUsdcWethPrime[0.001]
```

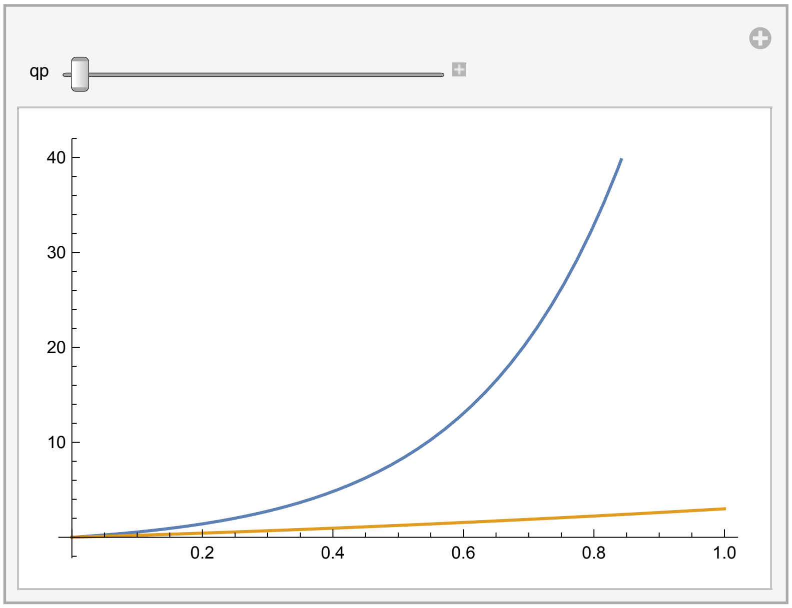
Out[46]= 1.10089

In[47]:=
```
Exp[lNormUsdcWethPrime[0.001]]
```
Out[47]=
```
3.00685
```
In[48]:=
```
Exp[-lNormUsdcWethPrime[0.001]]
```
Out[48]=
```
0.332573
```
In[49]:=
```
(* if take up 100% of cap,
will have 4x slippage to upside and -76% slippage to downside *)
```
In[50]:=
```
(* What's the slippage function? Compare vs Uniswap as well ... *)
```
In[51]:=
```
(* Us: P/P0 - 1 = e^{s+l*q} - 1; Uniswap: P/P0 - 1 = (1+q)^2 - 1 *)
```
In[52]:=
```
(* take s = 0 for here *)
```
In[53]:=
```
slippageNormUsdcsWeth[qp_, q_] := Exp[lNormUsdcWethPrime[qp] * q] - 1
```
In[54]:=
```
slippageNormWithSpreadUsdcsWeth[qp_, q_] :=
 Exp[s * Sign[q] + lNormUsdcWethPrime[qp] * q] - 1
```
In[55]:=
```
uniswapSlippageUp[q_] := (1 + q) ^2 - 1
```
In[56]:=
```
uniswapSlippageDown[q_] := 1 / (1 + q) ^2 - 1
```
In[57]:=
```
Manipulate[Plot[slippageNormUsdcsWeth[qp, q], {q, 0, 1}, PlotRange → All],
 {qp, 0.00025, 0.01}]
```
Out[57]=



In[58]:=
```
Manipulate[Plot[{slippageNormUsdcsWeth[qp, x], uniswapSlippageUp[x] },
  {x, 0, 1.0}], {qp, 0.00025, 0.01}]
```

Out[58]=



In[59]:=
```
(* Values that matter for
 us: q in [0, 1] where q is percentage of OI cap user takes up *)
```

In[60]:=
```
(* For values that matter,
can calibrate slippage such that scalp trade is -EV for traders taking up >
 0.1% of cap when price jumps *)
```

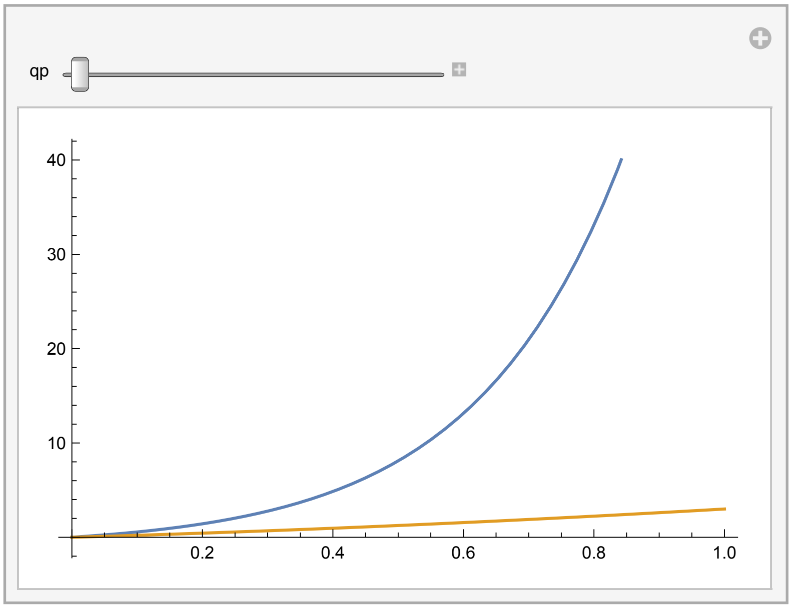
In[61]:=
```
(* We are the blue. Uniswap V2 x*y=
 k is the orange in the same region. We can have less slippage than
  Uniswap in all regions of interest. Chart above assumes our OI max is
  the same as 1/2 entire Uniswap liquidity (# of 'x' tokens in pool) *)
```

In[62]:=

In[63]:=
```
(* Compare with the static spread as well ... *)
```

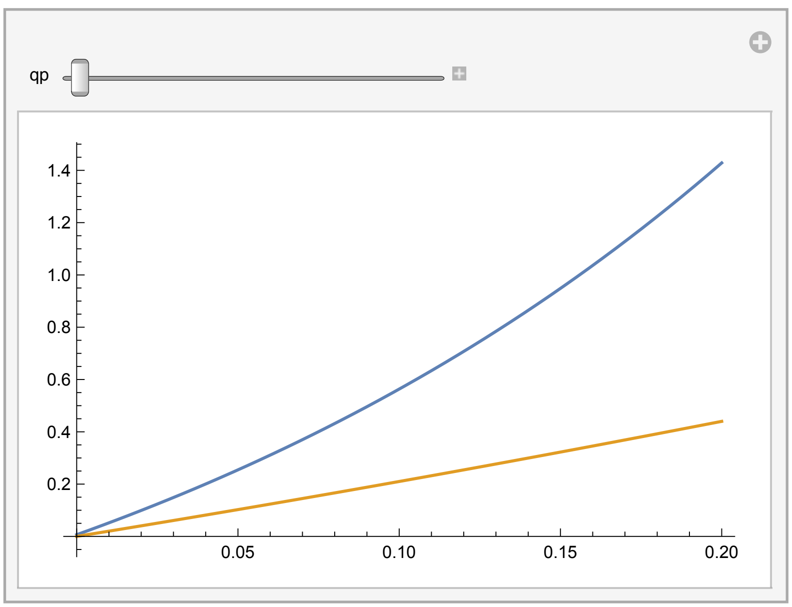
In[64]:=
```
Manipulate[Plot[{slippageNormWithSpreadUsdcsWeth[qp, x], uniswapSlippageUp[x] },
  {x, 0, 1.0}], {qp, 0.00025, 0.01}]
```

Out[64]=



In[65]:=
```
Manipulate[Plot[{slippageNormWithSpreadUsdcsWeth[qp, x], uniswapSlippageUp[x] },
  {x, 0, 0.2}], {qp, 0.00025, 0.01}]
```

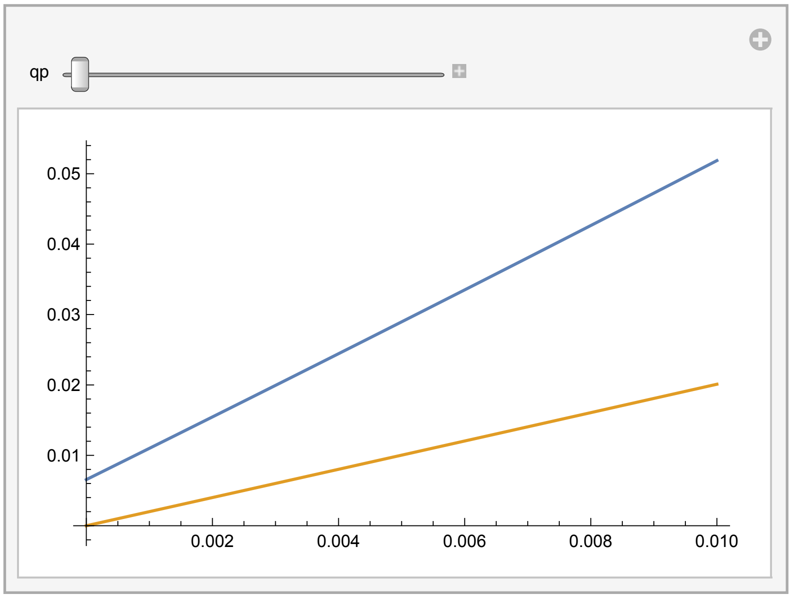
Out[65]=



In[66]:=
```
(* Still beautiful. Zoom in around x = 0 *)
```

In[67]:=
```
Manipulate[Plot[{slippageNormWithSpreadUsdcsWeth[qp, x], uniswapSlippageUp[x] },
  {x, 0, 0.01}], {qp, 0.00025, 0.01}]
```

Out[67]=

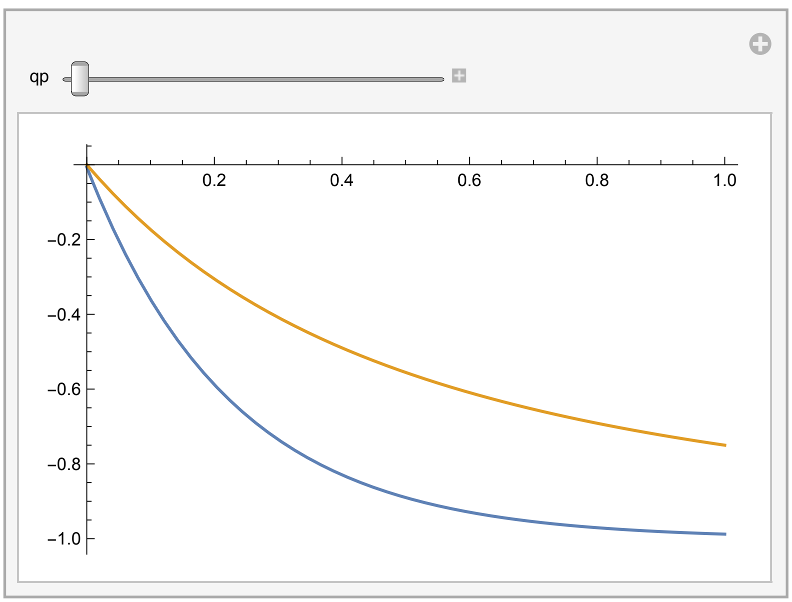


In[68]:=
```
(* And the static spread is apparent :) *)
```

In[69]:=
```
(* What about slippage on the downside? *)
```

In[70]:=
```
Manipulate[
 Plot[{slippageNormWithSpreadUsdcsWeth[qp, -x], uniswapSlippageDown[x] },
  {x, 0, 1.0}], {qp, 0.00025, 0.01}]
```

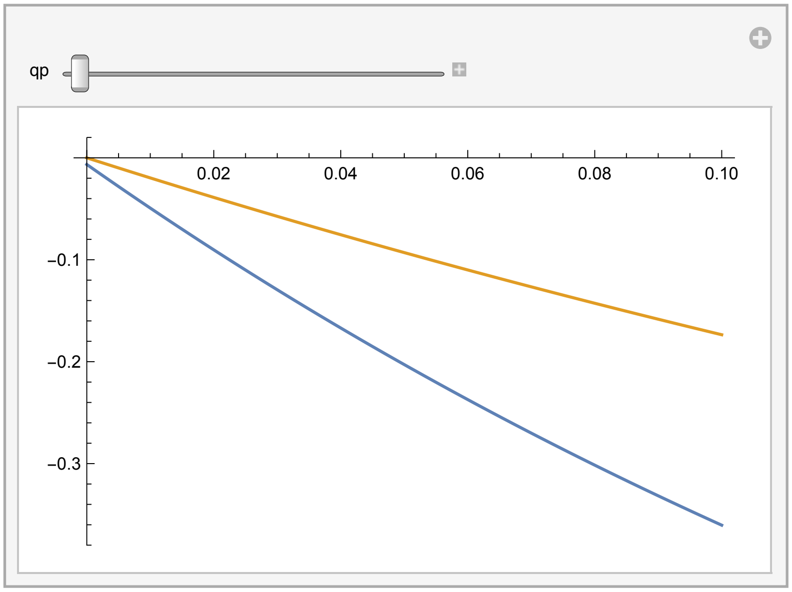
Out[70]=



In[71]:=
```
(* Beautiful here as well ... *)
```

In[72]:=
```
Manipulate[
 Plot[{slippageNormWithSpreadUsdcsWeth[qp, -x], uniswapSlippageDown[x] },
  {x, 0, 0.1}], {qp, 0.00025, 0.01}]
```

Out[72]=



In[73]:=
```
(* Now: 1. check w stable distribution math + caps;
2. Can a user profitably chunk the
  trade over the next 40 blocks to get EV ≥ 0? *)
```

In[74]:=
```
(* From Mathematica fit ... *)
```

In[75]:=
```
edistWethUsdc90dFiltered10MinCandle
```

Out[75]=
```
StableDistribution[1, 1.46465, -0.0496207, -0.000010553, 0.00154277]
```

In[76]:=
```
cp = 4
```

Out[76]=
```
4
```

In[77]:=
```
Log[1.0 + cp]
```

Out[77]=
```
1.60944
```

In[78]:=
```
s
```

Out[78]=
```
0.00652983
```

In[79]:=
```
edistWethUsdcYv10MinCandle = StableDistribution[1, 1.4646547203677354`,
  -0.04962071457296463`, -0.000010553047315527044 - 2.0 * s, 0.0015427769619594313`]
```

Out[79]=
```
StableDistribution[1, 1.46465, -0.0496207, -0.0130702, 0.00154277]
```

In[80]:=
```
1 + cp
```

Out[80]=
```
5
```

In[81]:=
```
gInv = Log[1.0 + cp]
```
Out[81]= 1.60944

In[82]:=
```
1 - CDF[edistWethUsdcYv10MinCandle, 0]
```
Out[82]= 0.00933603

In[83]:=
```
CDF[edistWethUsdcYv10MinCandle, 0]
```
Out[83]= 0.990664

In[84]:=
```
(* Beautiful.... Less than normal *)
```

In[85]:=
```
1 - CDF[edistWethUsdcYv10MinCandle, gInv]
```
Out[85]= 7.48682×10^{-6}

In[86]:=
```
CDF[edistWethUsdcYv10MinCandle, gInv]
```
Out[86]= 0.999993

In[87]:=
```
(* This is to reach the 5x in 10m so negligible which is good *)
```

In[88]:=
```
1 - CDF[edistWethUsdcYv10MinCandle, 0] -
  (1 + cp) * (1 - CDF[edistWethUsdcYv10MinCandle, gInv])
```
Out[88]= 0.0092986

In[89]:=
```
NIntegrate[PDF[edistWethUsdcYv10MinCandle, y] * Exp[y], {y, 0, gInv}]
```
Out[89]= 0.00956781

In[90]:=
```
(NIntegrate[PDF[edistWethUsdcYv10MinCandle, y] * Exp[y], {y, 0, gInv}]) /
  (1 - CDF[edistWethUsdcYv10MinCandle, 0] -
    (1 + cp) * (1 - CDF[edistWethUsdcYv10MinCandle, gInv]))
```
Out[90]= 1.02895

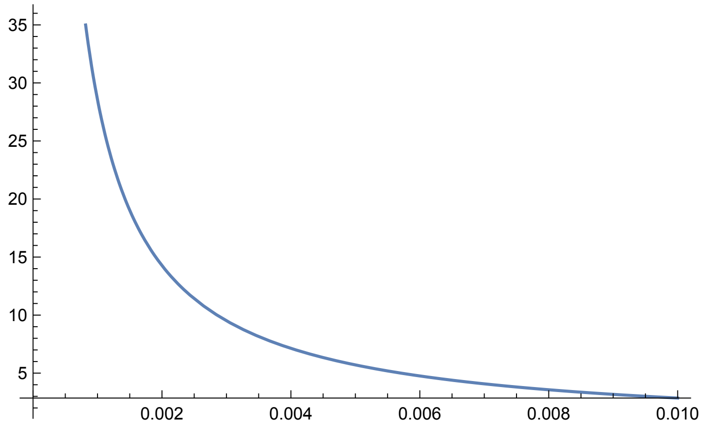
In[91]:=
```
logStableUsdcWeth =
  Log[(NIntegrate[PDF[edistWethUsdcYv10MinCandle, y] * Exp[y], {y, 0, gInv}]) /
    (1 - CDF[edistWethUsdcYv10MinCandle, 0] -
      (1 + cp) * (1 - CDF[edistWethUsdcYv10MinCandle, gInv]))]
```
Out[91]= 0.0285405

In[92]:=
```
rhs
```
Out[92]= 0.00110089

In[93]:=
```
(* way higher than EV from norm :)... 2.854% for stable vs 0.11% for norm *)
```

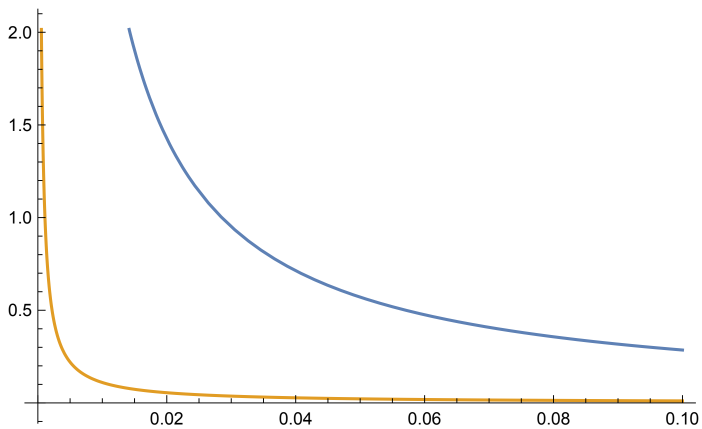
In[94]:=
```
lStableUsdcWethPrime[q_] := (1 / q) * logStableUsdcWeth
```

In[95]:=
```
Plot[lStableUsdcWethPrime[q], {q, 0, 0.01}]
```

Out[95]=


In[96]:=
```
(* Compare with normal ... *)
```

In[97]:=
```
Plot[{lStableUsdcWethPrime[q], lNormUsdcWethPrime[q]}, {q, 0, 0.1}]
```

Out[97]=


In[98]:=
```
lStableUsdcWethPrime[0.01]
```

Out[98]= 2.85405

In[99]:=
```
lNormUsdcWethPrime[0.01]
```

Out[99]= 0.110089

In[100]:=
```
lStableUsdcWethPrime[0.01] / lNormUsdcWethPrime[0.01] - 1
```

Out[100]= 24.9248

In[101]:=
```
(* 25x difference! in market impact
 parameter to obtain negative EV for 1% of OI cap *)
```

In[102]:=
```
(* 1% of OI for negative EV is likely what we want. Check slippage! *)
```

In[103]:=
```
Exp[lNormUsdcWethPrime[0.01]]
```

Out[103]= 1.11638

In[104]:=
```
Exp[lStableUsdcWethPrime[0.01]]
```
Out[104]= 17.3579

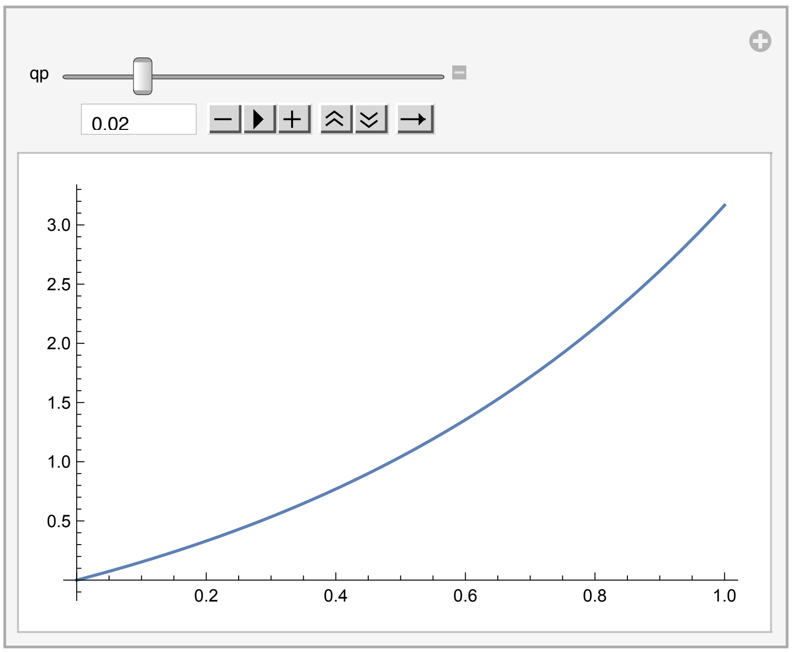
In[105]:=
```
Exp[-lStableUsdcWethPrime[0.01]]
```
Out[105]= 0.0576105

In[106]:=
```
slippageStableUsdcsWeth[qp_, q_] := Exp[lStableUsdcWethPrime[qp] * q] - 1
```

In[107]:=
```
slippageStableWithSpreadUsdcsWeth[qp_, q_] :=
 Exp[s * Sign[q] + lStableUsdcWethPrime[qp] * q] - 1
```

In[108]:=
```
Manipulate[Plot[slippageStableUsdcsWeth[qp, q], {q, 0, 1}, PlotRange → All],
 {qp, 0.0025, 0.1}]
```

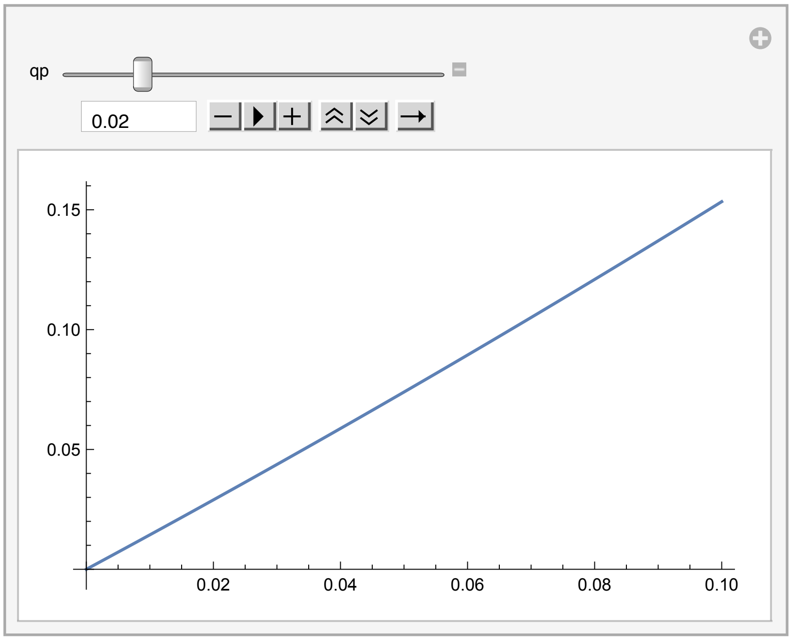
Out[108]=



In[109]:=
```
(* Zoom in for smaller fish trades ... *)
```

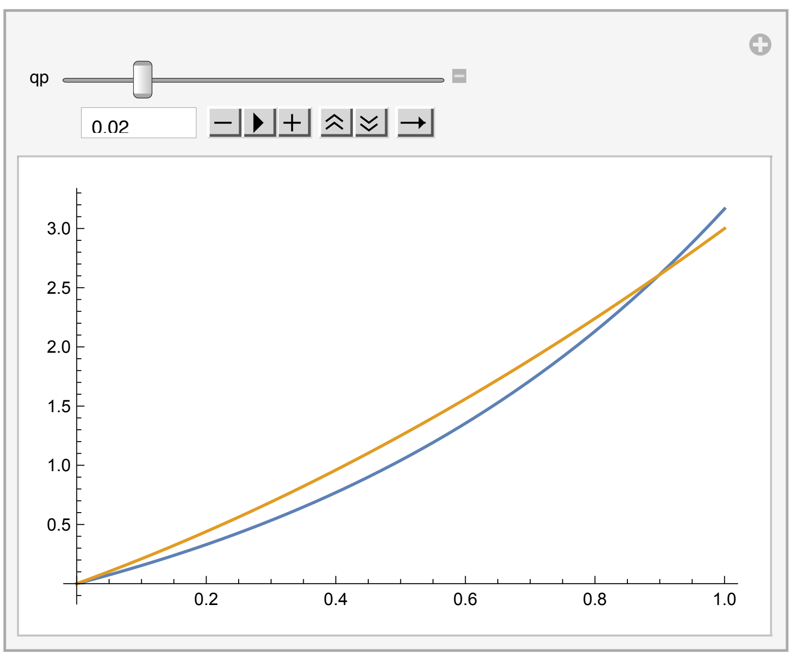
```
In[110]:= Manipulate[Plot[slippageStableUsdcsWeth[qp, q],
            {q, 0, 0.1}, PlotRange → All], {qp, 0.0025, 0.1}]
```

Out[110]=




```
In[111]:= (* Compare w Uniswap ... *)

In[112]:= Manipulate[Plot[{slippageStableUsdcsWeth[qp, x], uniswapSlippageUp[x] },
            {x, 0, 1.0}], {qp, 0.0025, 0.1}]
```

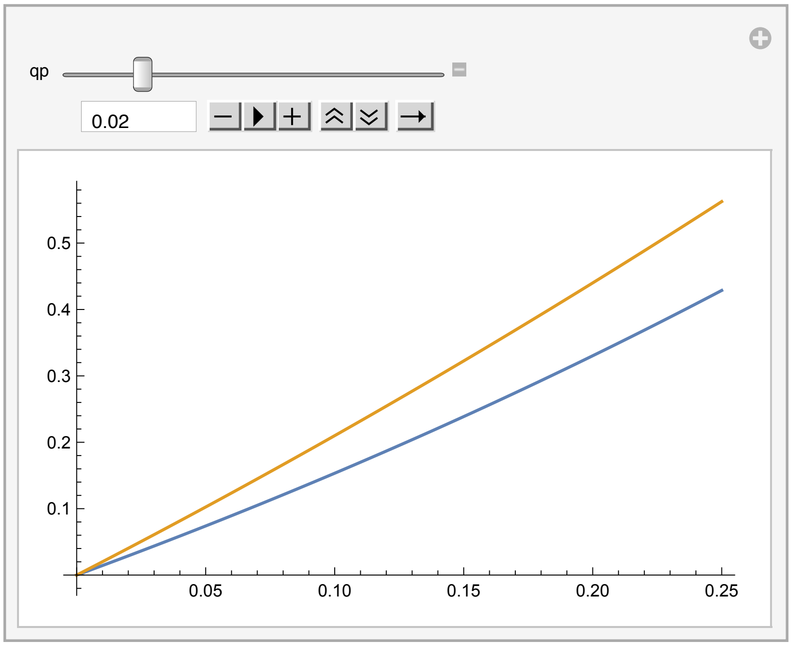
Out[112]=




```
In[113]:= (* Zoom in for small fish *)
```

In[114]:=
```
Manipulate[Plot[{slippageStableUsdcsWeth[qp, x], uniswapSlippageUp[x] },
  {x, 0, 0.25}], {qp, 0.0025, 0.1}]
```

Out[114]=

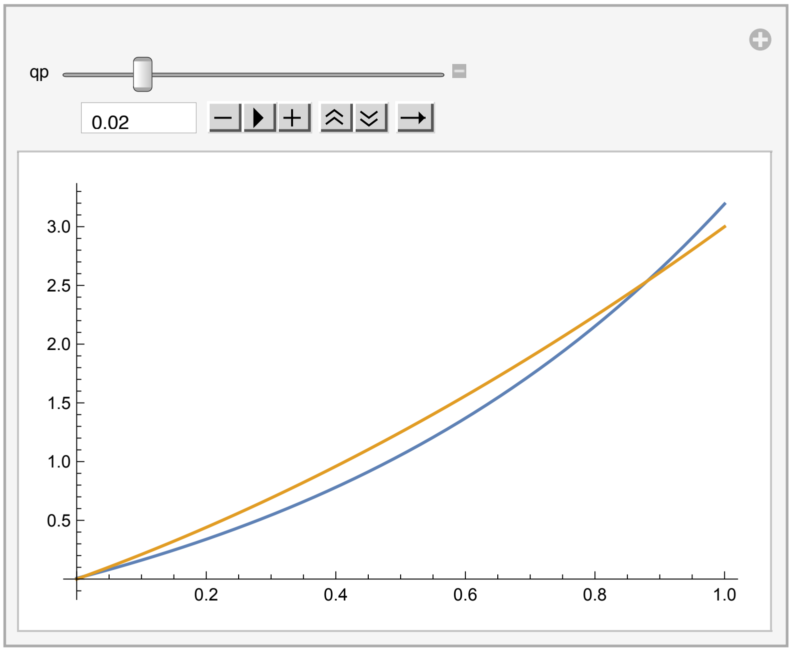


In[115]:=
```
(* If we risk 2% of the OI cap vs 1%,
we're at Uniswap slippage levels. This is beautiful *)
```

In[116]:=
```
(* Plot with static spread then check how much we can get milked for <=
   2% (worst case) ... max of EV given Q0 = 2% of Qmax .. *)
```

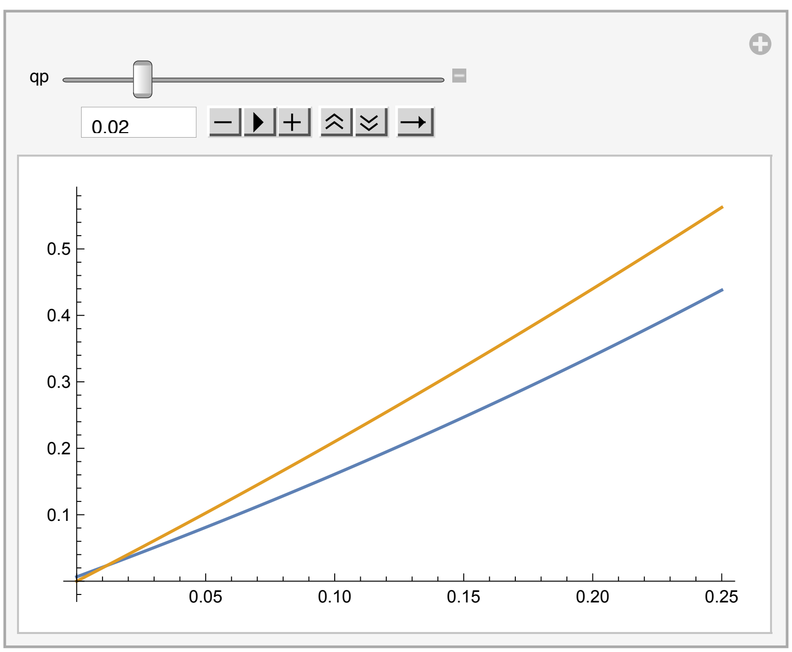
In[117]:=
```
Manipulate[Plot[{slippageStableWithSpreadUsdcsWeth[qp, x], uniswapSlippageUp[x] },
  {x, 0, 1.0}], {qp, 0.0025, 0.1}]
```

Out[117]=



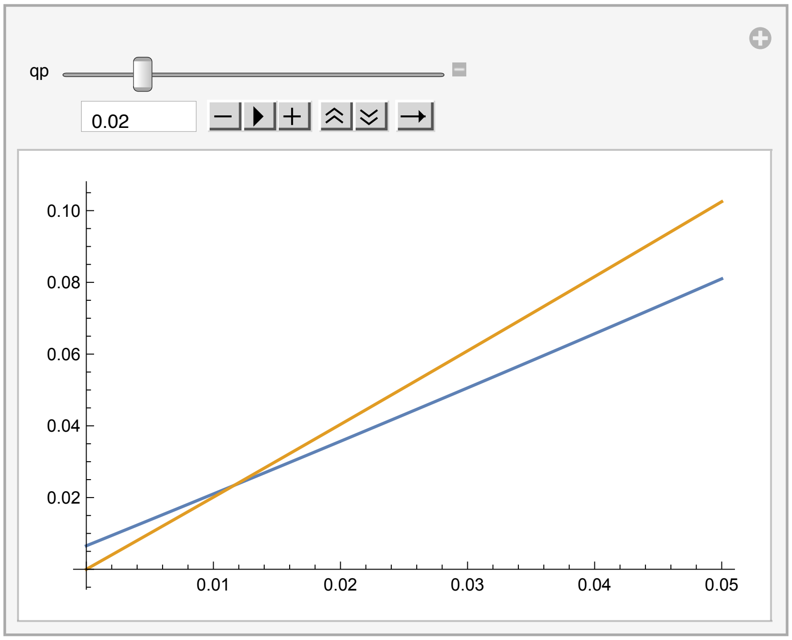
In[118]:=
```
Manipulate[Plot[{slippageStableWithSpreadUsdcsWeth[qp, x], uniswapSlippageUp[x] },
  {x, 0, 0.25}], {qp, 0.0025, 0.1}]
```

Out[118]=



In[119]:=
```
Manipulate[Plot[{slippageStableWithSpreadUsdcsWeth[qp, x], uniswapSlippageUp[x] },
  {x, 0, 0.05}], {qp, 0.0025, 0.1}]
```

Out[119]=



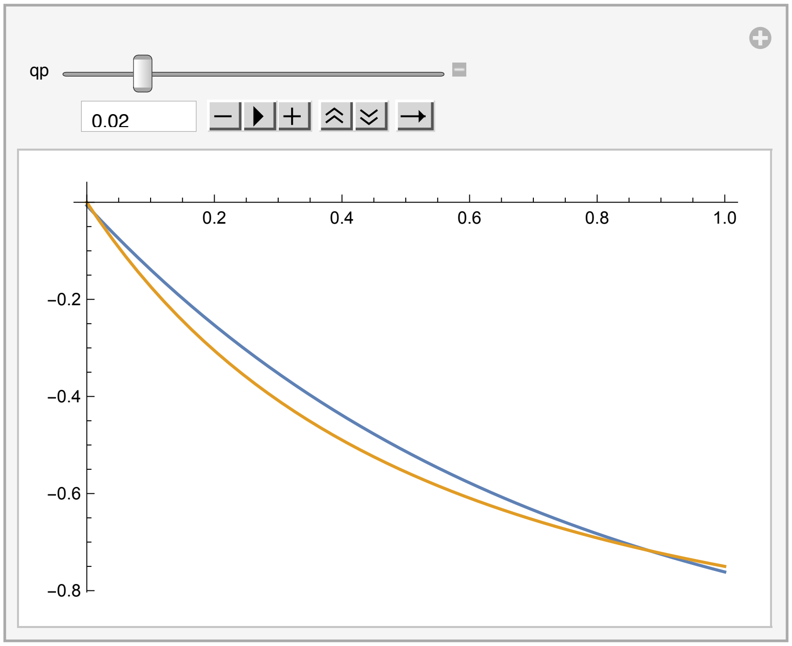
In[120]:=
```
slippageStableWithSpreadUsdcsWeth[0.02, 0.01]
```

Out[120]= 0.0210179

In[121]:=
```
(* And on the downside ... *)
```

In[122]:=
```
Manipulate[
 Plot[{slippageStableWithSpreadUsdcsWeth[qp, -x], uniswapSlippageDown[x] },
  {x, 0, 1.0}], {qp, 0.0025, 0.1}]
```

Out[122]=



In[123]:=
```
h = Log[NIntegrate[PDF[edistWethUsdcYv10MinCandle, y] * Exp[y], {y, 0, gInv}] /
    (1 - CDF[edistWethUsdcYv10MinCandle, 0])]
```

Out[123]= 0.0245228

In[124]:=
```
rho = (1 + cp) * (1 - CDF[edistWethUsdcYv10MinCandle, gInv]) /
    (1 - CDF[edistWethUsdcYv10MinCandle, 0])
```

Out[124]= 0.00400964

In[125]:=
```
Exp[h - lStableUsdcWethPrime[0.02] * 0.01] - 1 + rho
```

Out[125]= 0.0143149

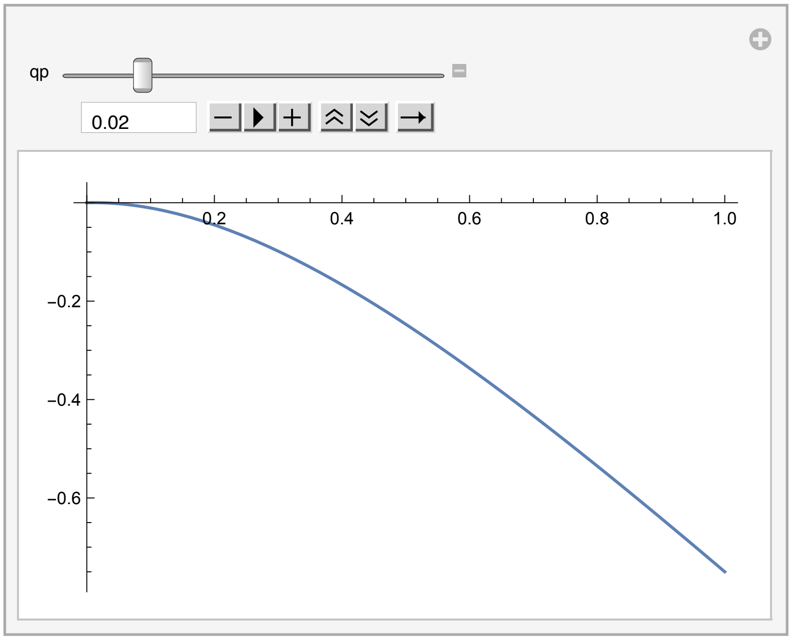
In[126]:=
```
Exp[h] - 1 + rho
```

Out[126]= 0.0288356

In[127]:=
```
evStableUsdcWeth[qp_, q_] := q * (Exp[h - lStableUsdcWethPrime[qp] * q] - 1 + rho)
```

In[128]:=
```
Manipulate[Plot[evStableUsdcWeth[qp, q], {q, 0, 1}], {qp, 0.0025, 0.1}]
```

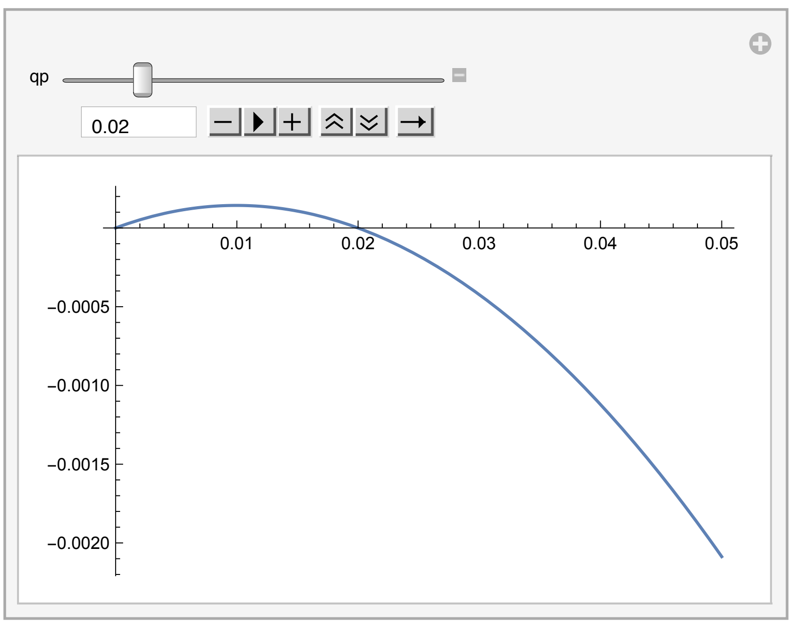
Out[128]=



In[129]:=
```
(* Zoom in where it really counts ... *)
```

In[130]:=
```
Manipulate[Plot[evStableUsdcWeth[qp, q], {q, 0, 0.05}], {qp, 0.0025, 0.1}]
```

Out[130]=



In[131]:=
```
(* EV max happens at 1/2 of Q0
  (same as norm). And for that 1% size on the 2% EV setting,
milk the system for about 0.000143 * OI cap => 1.43 bps of OI cap on the scalp *)
```

In[132]:=
```
evStableUsdcWeth[0.02, 0.01]
```

Out[132]= 0.000143149

```
In[133]:= evStableUsdcWeth[0.02, 0.01] / 0.01

Out[133]= 0.0143149

In[134]:= (* so for example, if the cap is 100M,
          we'd expect to get milked for max $14.3k on the scalp with size $1M. *)

In[135]:= 0.000143 * 100 000 000

Out[135]= 14 300.

In[136]:= (* Since fees are about 15bps on each
            side => so 30 bps total. Eats into ~20% of scalps profit ... *)

In[137]:= 0.0030 * 1 000 000

Out[137]= 3000.

In[138]:= 3000 / 14 300.0

Out[138]= 0.20979

In[139]:= (* How much more volume does it take to overcome that scalp if charging 30bps? *)

In[140]:= 14 300 - 3000.0

Out[140]= 11 300.

In[141]:= 11 300 / (0.0030)

Out[141]= 3.76667 × 10^6

In[142]:= (* $3.76M more in volume needed to recover on $1M scalp with
           $100M cap. Numbers scale accordingly. If we have caps at $100M OI,
          then likely doing way more than $3.76M in volume per day. Profitable
           scalp also happens only ~1% of the time, if fit params correctly *)

In[143]:= (* More realistically at launch, $10M OI cap,
          means $100k position size scalp with $1.4k in profit,
          1% of the time. Need volume of $376k to recover those funds, which is fine *)

In[144]:=

In[145]:= (* Example *)

In[146]:= (* Say 4% jump in a block, 1.3% killed by static,
          2.7% left to take as a scalp ... It takes 40 blocks
           to catch up to that 4% jump in spot. About linearly over 40
           blocks. so price received moves up about 0.1% per block. *)

In[147]:= slippageStableUsdcsWeth[0.02, 0.01]

Out[147]= 0.0143726
```

```
In[148]:= (* if it resets per block,
I can enter into this trade each block for 40 blocks (if I wish),
building up my position size *)

In[149]:= slippageStableUsdcsWeth[0.02, 0.01] + 2 * s

Out[149]= 0.0274322

In[150]:= (* So i get 1.3% first block in PnL on 1% of the OI, 1.2% second block
 in PnL on another 1% of the OI, ... 0% for the first 13 blocks *)

In[151]:= (* So i end up occupying 13% of the OI cap. with an average return of ... *)

In[152]:= .013 * 13 - 0.001 * 13 * 14 / 2.0

Out[152]= 0.078

In[153]:= (* 7.8% return on the 13% built up of the OI cap *)

In[154]:= (* Yea, we need to make Q rolling based
 on OI that has been queued in the last 10 minutes *)

In[155]:= (* TODO: Check out slippage amounts for something like RUNE-
  ETH or ALCX-ETH to see massive tail distr *)

In[156]:= (* ALCX-WETH 10m data: 'alpha:0.8608514241695278,beta:-0.11610545938130679,
mu:5.053317630867739e-05,sigma:0.00023687933293464885' *)

In[157]:= edistAlcxWeth90dFiltered10MinCandle = StableDistribution[1, 0.8608514241695278,
  -0.11610545938130679, 0.000050533176308677394`, 0.00023687933293464885]

Out[157]= StableDistribution[1, 0.860851, -0.116105, 0.0000505332, 0.000236879]

In[158]:= sAlcxWeth = 0.0061034683292931794

Out[158]= 0.00610347

In[159]:= (* This should be scary given alpha < 1 ... *)

In[160]:= edistAlcxWethYv10MinCandle =
 StableDistribution[1, 0.8608514241695278, -0.11610545938130679,
  0.000050533176308677394 - 2 * sAlcxWeth, 0.00023687933293464885]

Out[160]= StableDistribution[1, 0.860851, -0.116105, -0.0121564, 0.000236879]

In[161]:= (NIntegrate[PDF[edistAlcxWethYv10MinCandle, y] * Exp[y], {y, 0, gInv}]) /
 (1 - CDF[edistAlcxWethYv10MinCandle, 0] -
   (1 + cp) * (1 - CDF[edistAlcxWethYv10MinCandle, gInv]))

Out[161]= 1.15415

In[162]:= Log[1.154152144302166`]

Out[162]= 0.143366
```

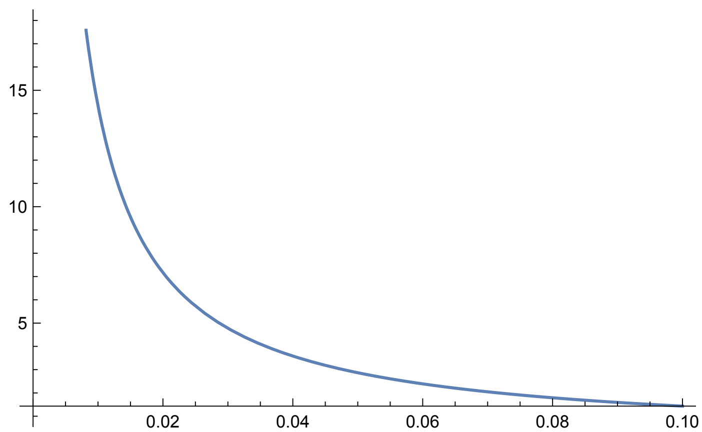
In[163]:=
```
(NIntegrate[PDF[edistWethUsdcYv10MinCandle, y] * Exp[y], {y, 0, gInv}]) /
 (1 - CDF[edistWethUsdcYv10MinCandle, 0] -
   (1 + cp) * (1 - CDF[edistWethUsdcYv10MinCandle, gInv]))
```
Out[163]= 1.02895

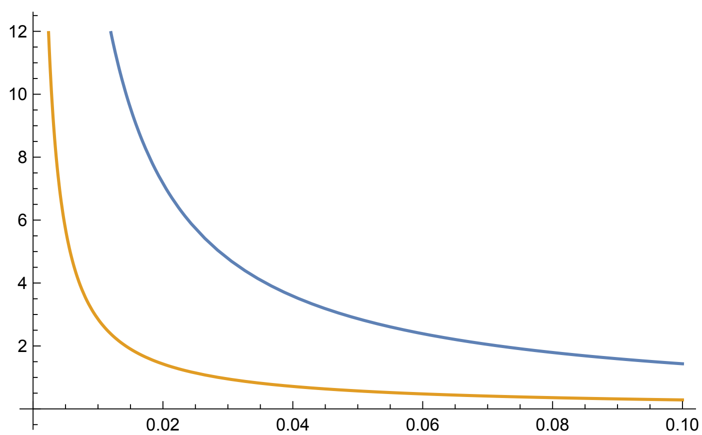
In[164]:=
```
Log[1.0289516801400798`]
```
Out[164]= 0.0285405

In[165]:=
```
(* Beautiful. 7x worse slippage for ALCX-WETH vs USDC-WETH. We want this *)
```

In[166]:=
```
logStableAlcxWeth =
 Log[(NIntegrate[PDF[edistAlcxWethYv10MinCandle, y] * Exp[y], {y, 0, gInv}]) /
   (1 - CDF[edistAlcxWethYv10MinCandle, 0] -
     (1 + cp) * (1 - CDF[edistAlcxWethYv10MinCandle, gInv]))]
```
Out[166]= 0.143366

In[167]:=
```
lStableAlcxWethPrime[q_] := (1 / q) * logStableAlcxWeth
```

In[168]:=
```
Plot[lStableAlcxWethPrime[q], {q, 0, 0.1}]
```
Out[168]=


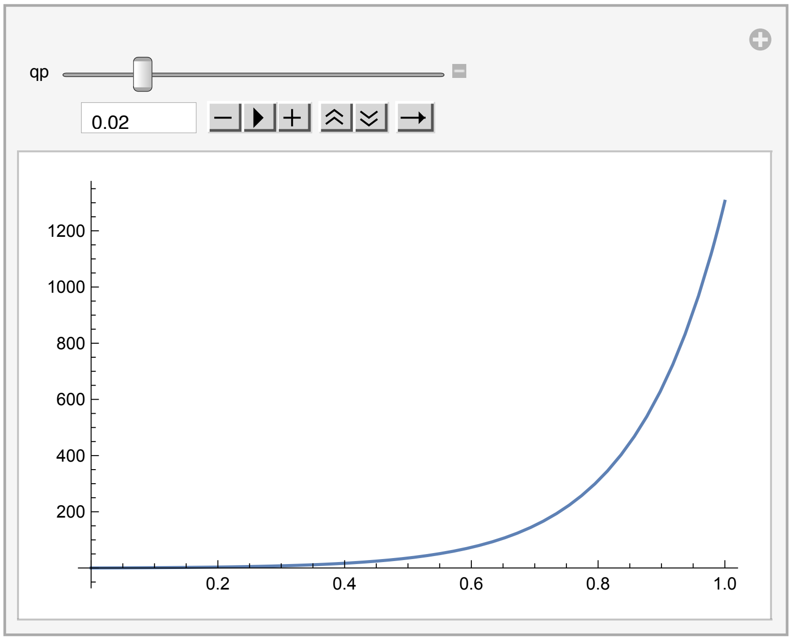
In[169]:=
```
Plot[{lStableAlcxWethPrime[q], lStableUsdcWethPrime[q]}, {q, 0, 0.1}]
```
Out[169]=


In[170]:=
```
(* And slippage ... *)
```

```
In[171]:= edistWethUsdc90dFiltered10MinCandle
Out[171]= StableDistribution[1, 1.46465, -0.0496207, -0.000010553, 0.00154277]

In[172]:= InverseCDF[edistWethUsdc90dFiltered10MinCandle , 0.995] / 2
Out[172]= 0.0097996

In[173]:= InverseCDF[edistAlcxWeth90dFiltered10MinCandle , 0.995] / 2
Out[173]= 0.0137858

In[174]:= (* 99.5% isn't terrible for delta static spread ...
             and now we have higher static for alcx as well ...
           but likely want the 99% mark so not as bad up front *)

In[175]:= (* Look at slippage at 99% level for ALCX *)

In[176]:= lStableAlcxWethPrime[0.02]
Out[176]= 7.1683

In[177]:= lStableUsdcWethPrime[0.02]
Out[177]= 1.42702

In[178]:= lStableAlcxWethPrime[0.02] / lStableUsdcWethPrime[0.02]
Out[178]= 5.02325

In[179]:= (* Cool, 5x worse slippage *)

In[180]:= slippageStableAlcxWeth[qp_, q_] := Exp[lStableAlcxWethPrime[qp] * q] - 1

In[181]:= slippageStableWithSpreadAlcxWeth[qp_, q_] :=
           Exp[s * Sign[q] + lStableAlcxWethPrime[qp] * q] - 1
```

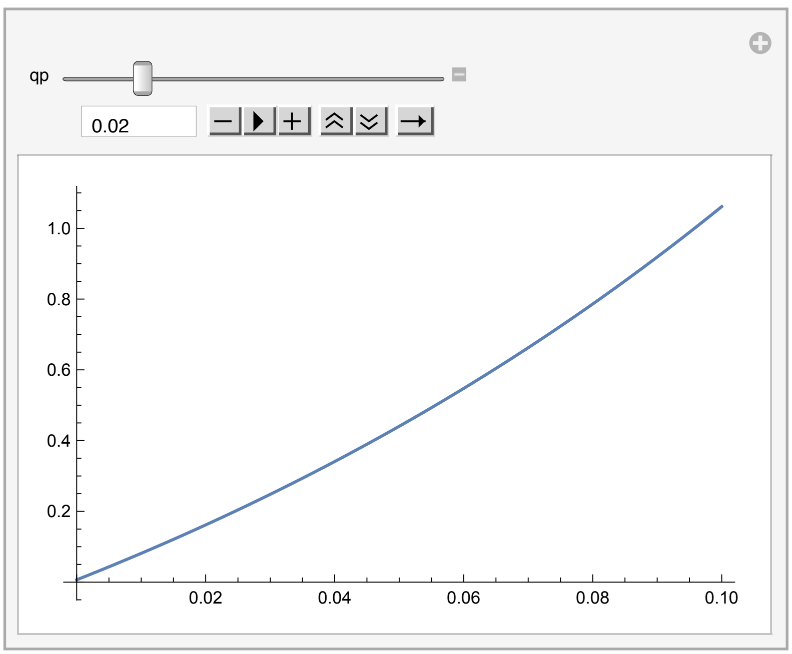
In[182]:=
```
Manipulate[Plot[slippageStableWithSpreadAlcxWeth[qp, q],
  {q, 0, 1}, PlotRange → All], {qp, 0.0025, 0.1}]
```

Out[182]=



In[183]:=
```
Manipulate[Plot[slippageStableWithSpreadAlcxWeth[qp, q],
  {q, 0, 0.1}, PlotRange → All], {qp, 0.0025, 0.1}]
```

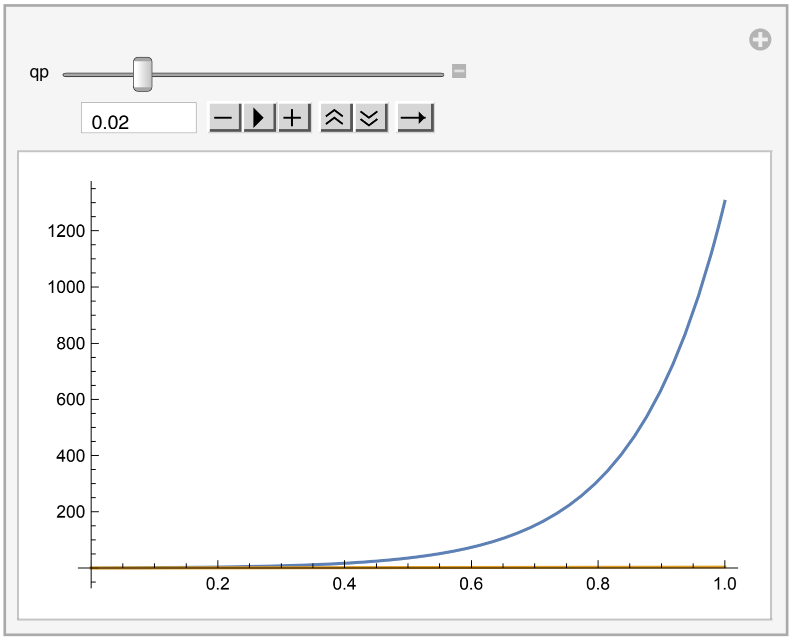
Out[183]=



In[184]:=
```
(* Compare w Uniswap slippage *)
```

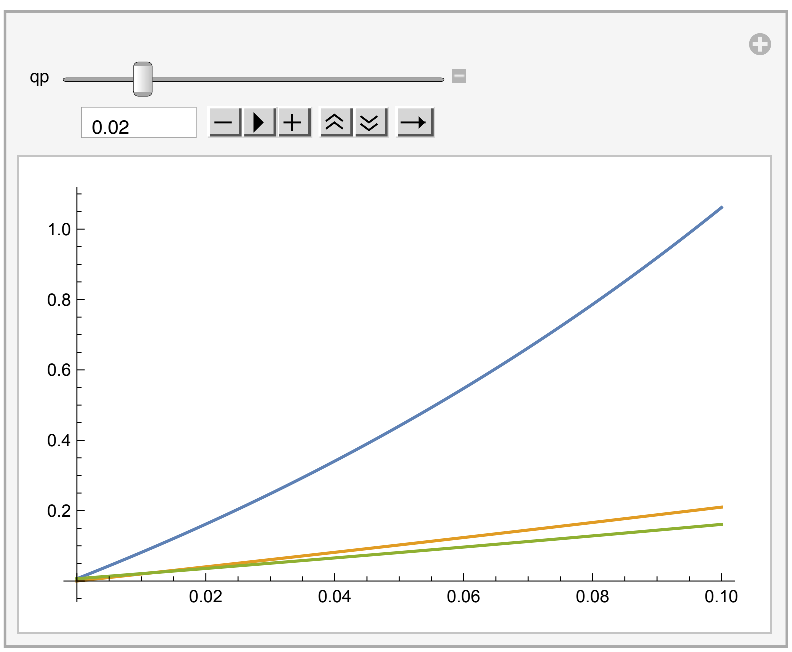
In[185]:=
```
Manipulate[Plot[{slippageStableWithSpreadAlcxWeth[qp, q], uniswapSlippageUp[q]},
  {q, 0, 1}, PlotRange → All], {qp, 0.0025, 0.1}]
```

Out[185]=



In[186]:=
```
Manipulate[Plot[{slippageStableWithSpreadAlcxWeth[qp, q],
   uniswapSlippageUp[q], slippageStableWithSpreadUsdcsWeth[qp, q]},
  {q, 0, 0.1}, PlotRange → All], {qp, 0.0025, 0.1}]
```

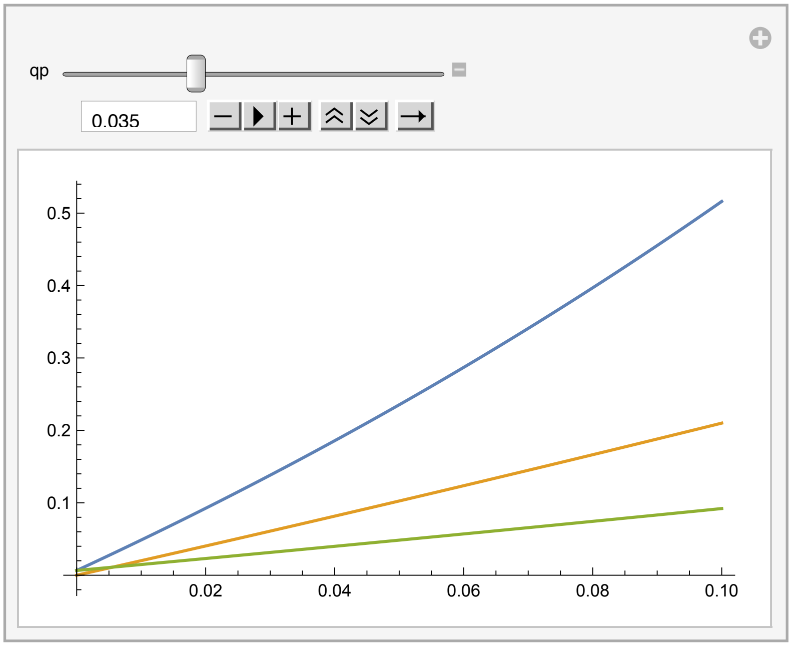
Out[186]=



In[187]:=
```
(* 2% seems pretty intense in terms of slippage. Will people just not trade? *)
```

In[188]:=
```
Manipulate[Plot[{slippageStableWithSpreadAlcxWeth[qp, q],
    uniswapSlippageUp[q], slippageStableWithSpreadUsdcsWeth[qp, q]},
  {q, 0, 0.1}, PlotRange → All], {qp, 0.0025, 0.1}]
```

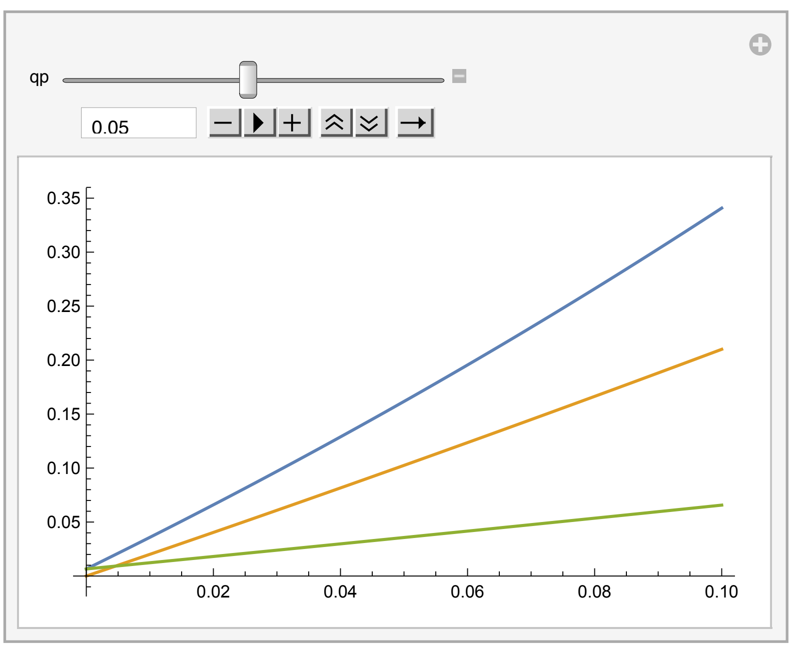
Out[188]=



In[189]:=
```
(* 3.5% is def better *)
```

In[190]:=
```
Manipulate[Plot[{slippageStableWithSpreadAlcxWeth[qp, q],
    uniswapSlippageUp[q], slippageStableWithSpreadUsdcsWeth[qp, q]},
  {q, 0, 0.1}, PlotRange → All], {qp, 0.0025, 0.1}]
```

Out[190]=

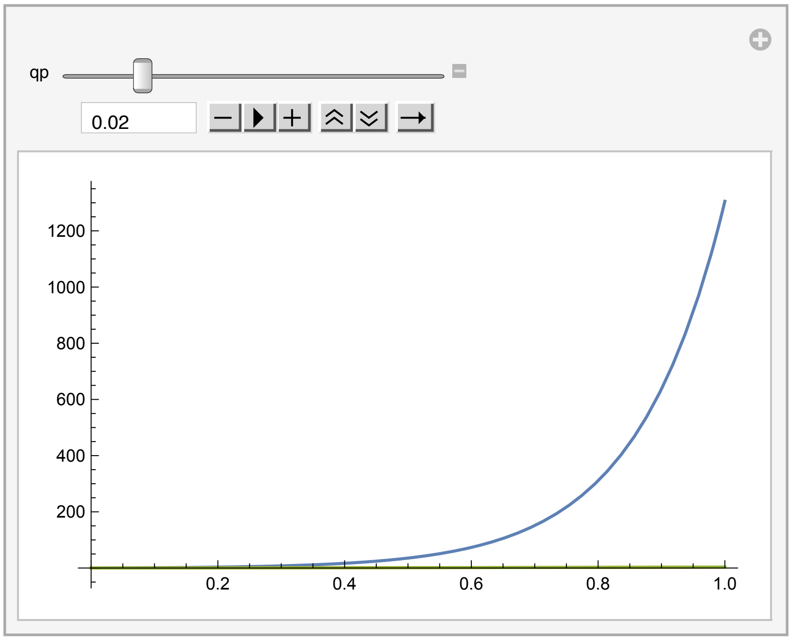


In[191]:=
```
(* 5% isn't terrible here as OI barrier,
when thinking in terms of slippage. What's the EV max for each? *)
```

In[192]:=
```
(* Zoom out again on upside then do downside ... *)
```

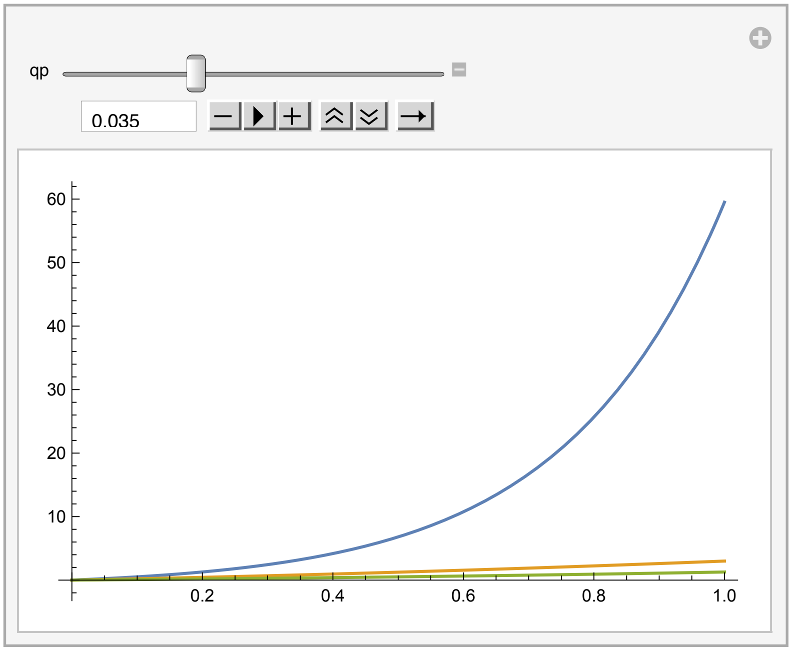
In[193]:=
```
Manipulate[Plot[{slippageStableWithSpreadAlcxWeth[qp, q],
    uniswapSlippageUp[q], slippageStableWithSpreadUsdcsWeth[qp, q]},
  {q, 0, 1}, PlotRange → All], {qp, 0.0025, 0.1}]
```

Out[193]=



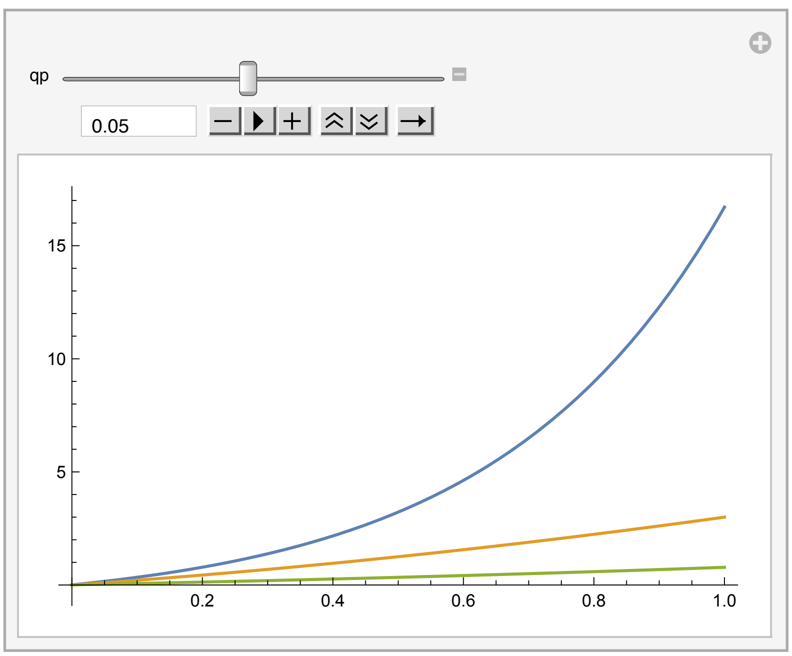
In[194]:=
```
Manipulate[Plot[{slippageStableWithSpreadAlcxWeth[qp, q],
    uniswapSlippageUp[q], slippageStableWithSpreadUsdcsWeth[qp, q]},
  {q, 0, 1}, PlotRange → All], {qp, 0.0025, 0.1}]
```

Out[194]=



In[195]:=
```
Manipulate[Plot[{slippageStableWithSpreadAlcxWeth[qp, q],
    uniswapSlippageUp[q], slippageStableWithSpreadUsdcsWeth[qp, q]},
  {q, 0, 1}, PlotRange → All], {qp, 0.0025, 0.1}]
```

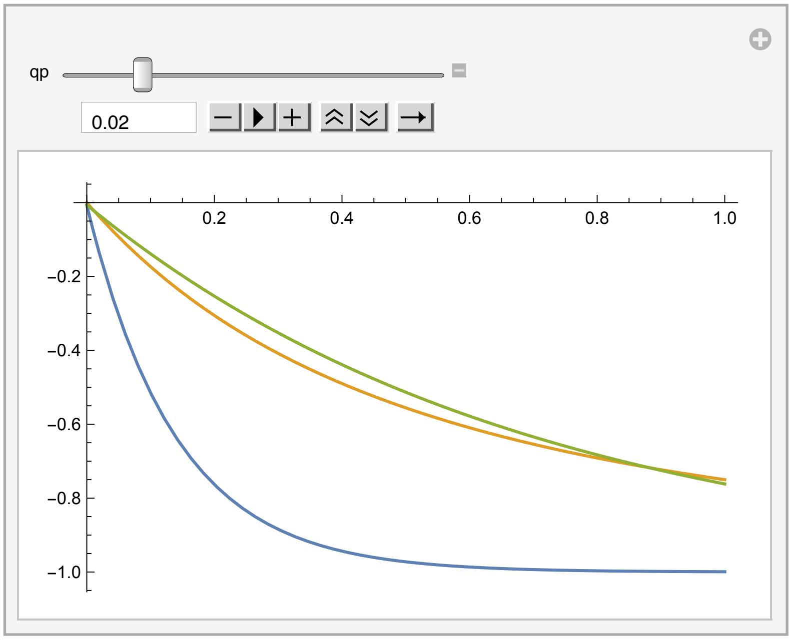
Out[195]=



In[196]:=
```
(* Do downside slippage *)
```

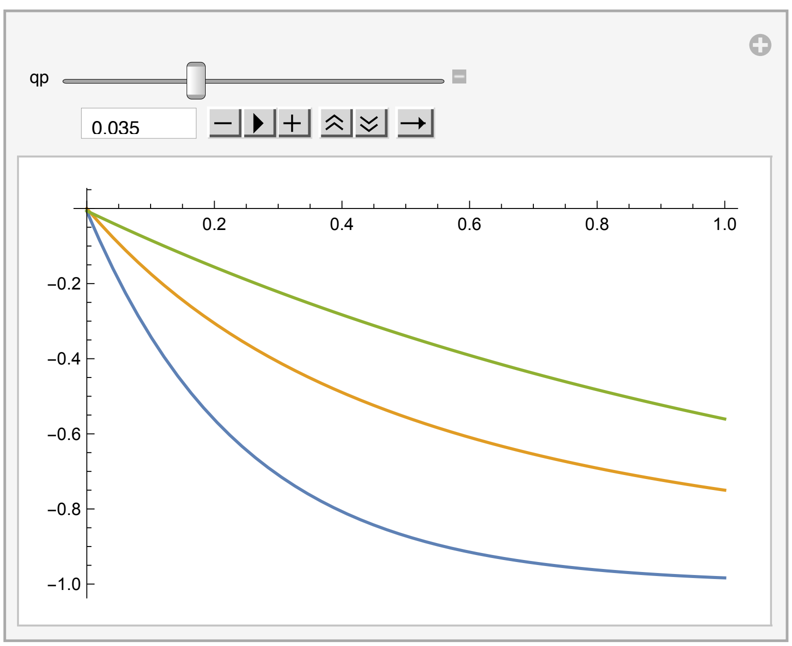
In[197]:=
```
Manipulate[
 Plot[{slippageStableWithSpreadAlcxWeth[qp, -x], uniswapSlippageDown[x] ,
   slippageStableWithSpreadUsdcsWeth[qp, -x]}, {x, 0, 1.0}], {qp, 0.0025, 0.1}]
```

Out[197]=



In[198]:=
```
Manipulate[
 Plot[{slippageStableWithSpreadAlcxWeth[qp, -x], uniswapSlippageDown[x] ,
   slippageStableWithSpreadUsdcsWeth[qp, -x]}, {x, 0, 1.0}], {qp, 0.0025, 0.1}]
```

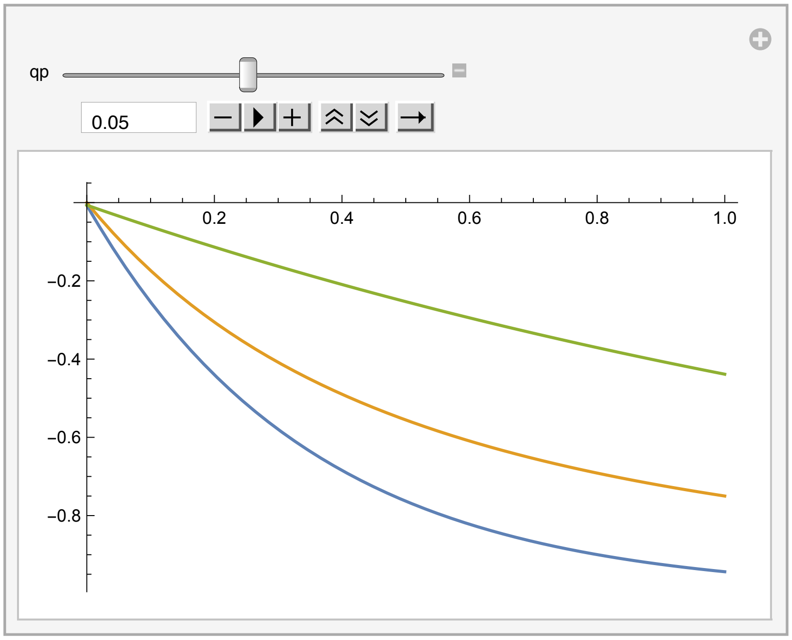
Out[198]=



In[199]:=
```
(* Honestly the 3.5% mark isn't terrible and still intense where it counts *)
```

In[200]:=
```
Manipulate[
 Plot[{slippageStableWithSpreadAlcxWeth[qp, -x], uniswapSlippageDown[x] ,
   slippageStableWithSpreadUsdcsWeth[qp, -x]}, {x, 0, 1.0}], {qp, 0.0025, 0.1}]
```

Out[200]=



In[201]:=
```
(* Ok, EV ... *)
```

In[202]:=
```
hAlcxWeth =
 Log[NIntegrate[PDF[edistAlcxWethYv10MinCandle, y] * Exp[y], {y, 0, gInv}] /
   (1 - CDF[edistAlcxWethYv10MinCandle, 0])]
```

Out[202]= 0.0648113

In[203]:=
```
rhoAlcxWeth = (1 + cp) * (1 - CDF[edistAlcxWethYv10MinCandle, gInv]) /
   (1 - CDF[edistAlcxWethYv10MinCandle, 0])
```

Out[203]= 0.0755485

In[204]:=
```
Exp[hAlcxWeth - lStableAlcxWethPrime[0.02] * 0.02 / 2.0] - 1 + rhoAlcxWeth
```

Out[204]= 0.0687004

In[205]:=
```
Exp[hAlcxWeth - lStableAlcxWethPrime[0.035] * 0.035 / 2.0] - 1 + rhoAlcxWeth
```

Out[205]= 0.0687004

In[244]:=
```
Exp[hAlcxWeth - lStableAlcxWethPrime[0.035] * 0.035] - 1 + rhoAlcxWeth
```

Out[244]= 2.77556×10^{-17}

In[206]:=
```
Exp[hAlcxWeth - lStableAlcxWethPrime[0.05] * 0.05 / 2.0] - 1 + rhoAlcxWeth
```

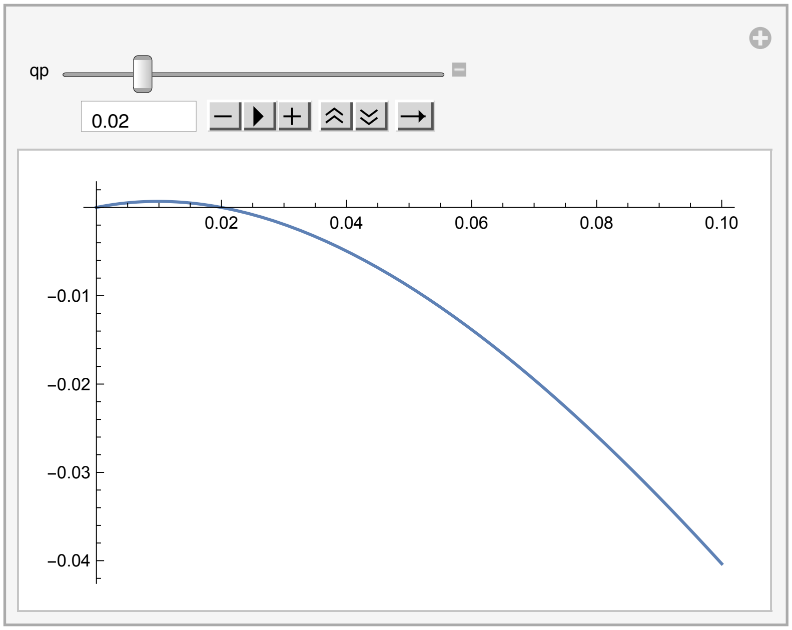
Out[206]= 0.0687004

In[207]:=
```
Exp[hAlcxWeth] - 1 + rhoAlcxWeth
```
Out[207]=
```
0.142506
```
In[208]:=
```
(* EV is 14.2% in 10m without the market
 impact. Max EV regardless of Q0 target is 6.87% scalp  *)
```
In[209]:=
```
(* Compare with usdc-weth... *)
```
In[210]:=
```
Exp[h] - 1 + rho
```
Out[210]=
```
0.0288356
```
In[211]:=
```
0.14250617688886105` / 0.028835594334819872`
```
Out[211]=
```
4.94202
```
In[212]:=
```
(* Good. 5x more in EV as well *)
```
In[213]:=
```
Exp[h - lStableUsdcWethPrime[0.02] * 0.02 / 2.0] - 1 + rho
```
Out[213]=
```
0.0143149
```
In[214]:=
```
Exp[h - lStableUsdcWethPrime[0.035] * 0.035 / 2.0] - 1 + rho
```
Out[214]=
```
0.0143149
```
In[215]:=
```
evStableUsdcWeth[0.02, 0.02 / 2.0] / (0.02 / 2.0)
```
Out[215]=
```
0.0143149
```
In[216]:=
```
evStableUsdcWeth[0.035, 0.035 / 2.0] / (0.035 / 2.0)
```
Out[216]=
```
0.0143149
```
In[217]:=
```
(* Nice *)
```
In[218]:=

In[219]:=
```
(* Look at ev function now *)
```
In[220]:=
```
evStableAlcxWeth[qp_, q_] :=
 q * (Exp[hAlcxWeth - lStableAlcxWethPrime[qp] * q] - 1 + rhoAlcxWeth)
```

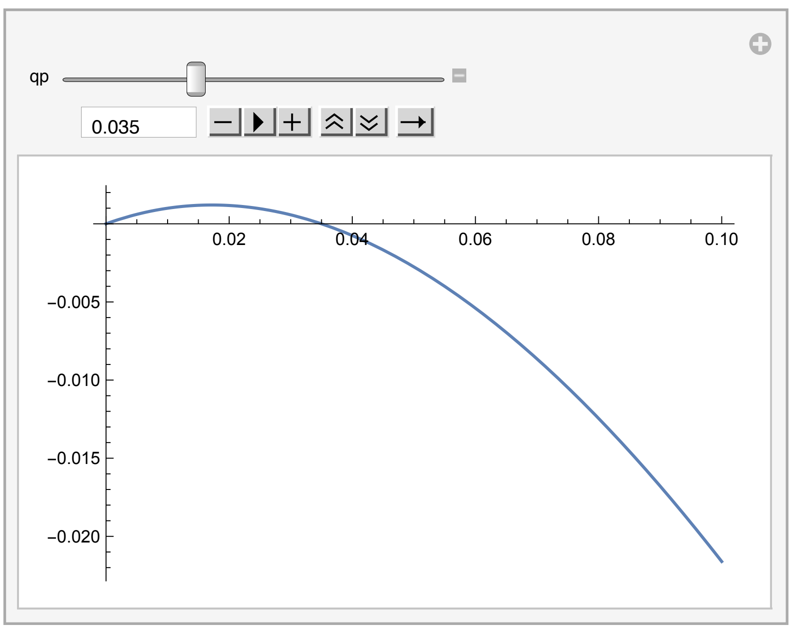
In[221]:=
```
Manipulate[Plot[evStableAlcxWeth[qp, q], {q, 0, 0.1}], {qp, 0.0025, 0.1}]
```

Out[221]=



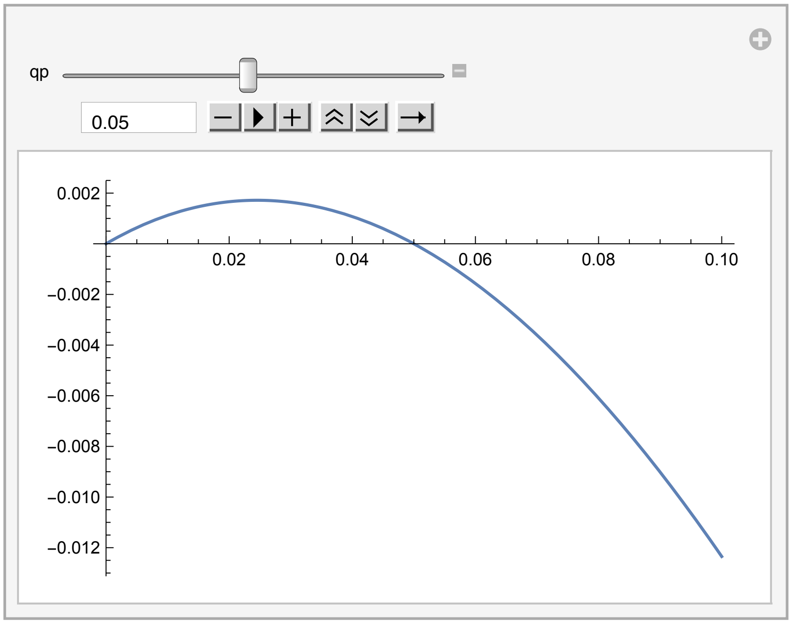
In[222]:=
```
Manipulate[Plot[evStableAlcxWeth[qp, q], {q, 0, 0.1}], {qp, 0.0025, 0.1}]
```

Out[222]=



In[223]:=
```
Manipulate[Plot[evStableAlcxWeth[qp, q], {q, 0, 0.1}], {qp, 0.0025, 0.1}]
```

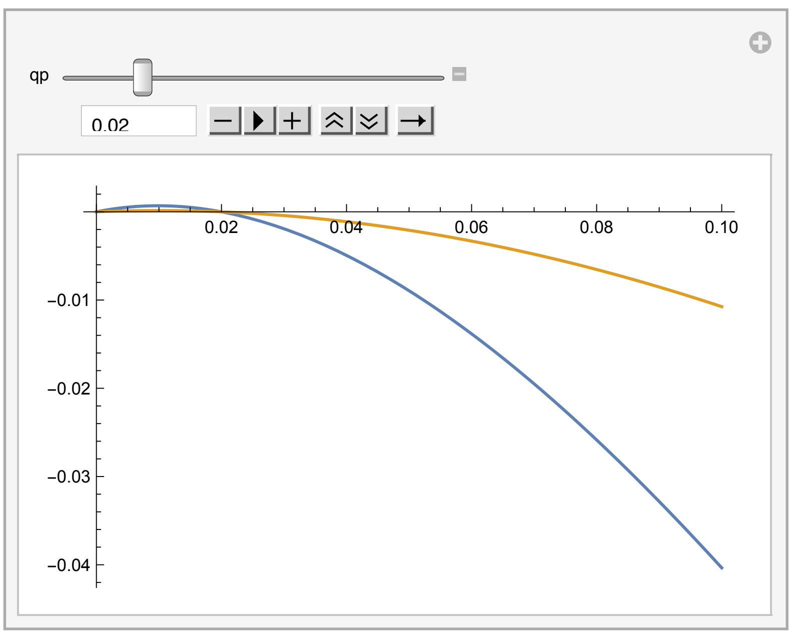
Out[223]=



In[224]:=
```
(* vs ev from Usdc Weth *)
```

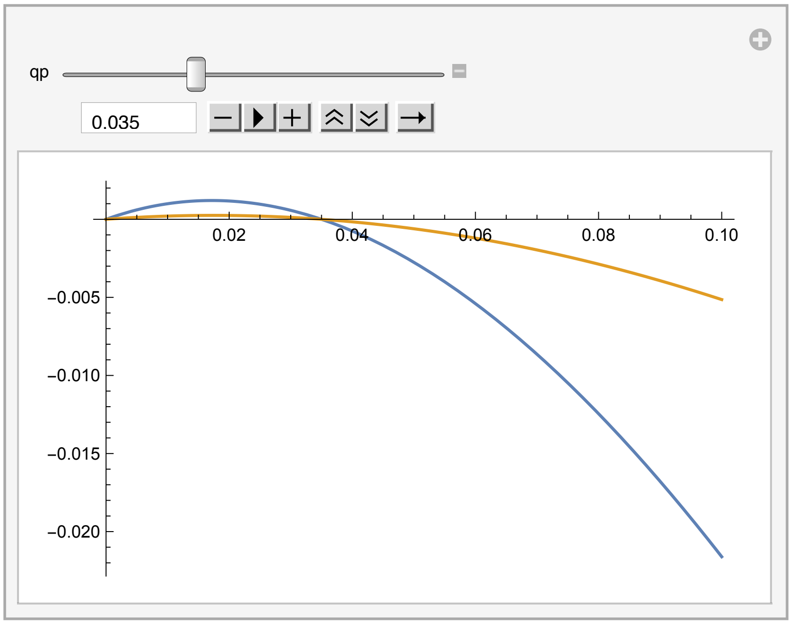
In[225]:=
```
Manipulate[Plot[{evStableAlcxWeth[qp, q], evStableUsdcWeth[qp, q]}, {q, 0, 0.1}],
 {qp, 0.0025, 0.1}]
```

Out[225]=



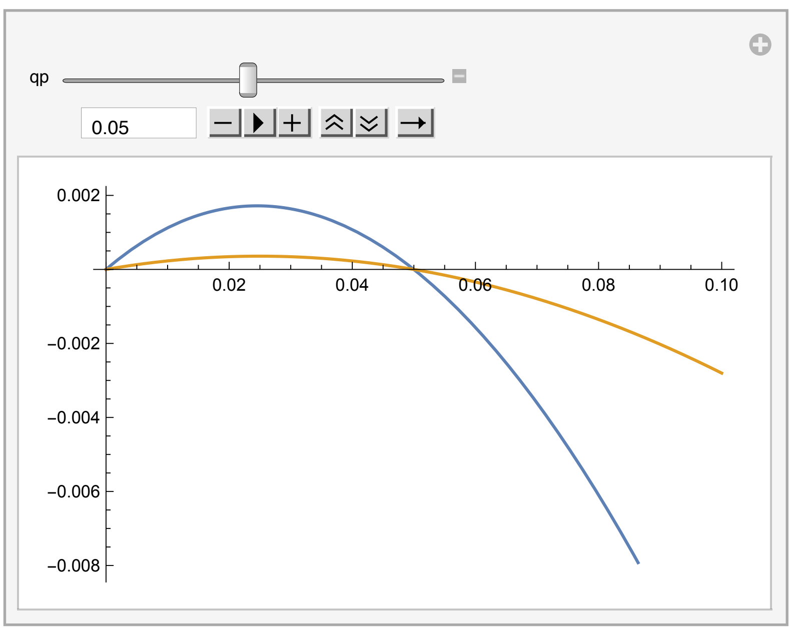
In[226]:=
```
Manipulate[Plot[{evStableAlcxWeth[qp, q], evStableUsdcWeth[qp, q]}, {q, 0, 0.1}],
 {qp, 0.0025, 0.1}]
```

Out[226]=



In[227]:=
```
Manipulate[Plot[{evStableAlcxWeth[qp, q], evStableUsdcWeth[qp, q]}, {q, 0, 0.1}],
 {qp, 0.0025, 0.1}]
```

Out[227]=



In[228]:=
```
(* So definitely risking more of the OI cap with ALCX-
 WETH than USDC-WETH. Likely need cap to be lower in realtive terms
  to compensate for it while still having comparable slippage. *)
```

In[229]:=
```
evStableAlcxWeth[0.02, 0.02 / 2.0]
```
Out[229]=
```
0.000687004
```
In[230]:=
```
evStableAlcxWeth[0.035, 0.035 / 2.0]
```
Out[230]=
```
0.00120226
```
In[231]:=
```
evStableAlcxWeth[0.05, 0.05 / 2.0]
```
Out[231]=
```
0.00171751
```
In[232]:=
```
(* Basically do we want to get scalped at 7bps of cap,
12bps of cap or 18bps of cap 1% of the time *)
```
In[233]:=
```
(* Compare w usdc weth again *)
```
In[234]:=
```
evStableUsdcWeth[0.02, 0.02 / 2.0]
```
Out[234]=
```
0.000143149
```
In[235]:=
```
evStableUsdcWeth[0.035, 0.035 / 2.0]
```
Out[235]=
```
0.000250511
```
In[236]:=
```
evStableUsdcWeth[0.05, 0.05 / 2.0]
```
Out[236]=
```
0.000357873
```
In[237]:=
```
(* For usdc weth, basically do we want to get scalped at 1.43bps of cap,
2.5bps of cap or 3.5bps of cap 1% of the time *)
```
In[238]:=
```
(* Likely would need cap on alcx-weth to be 5x smaller than cap on USDC-
 WETH then. If using CDF for caps as well,
this should come in similarly in derivation *)
```
In[239]:=
```
(* Think about optimization problem bw slippage vs EV,
instead of manual approach here *)
```