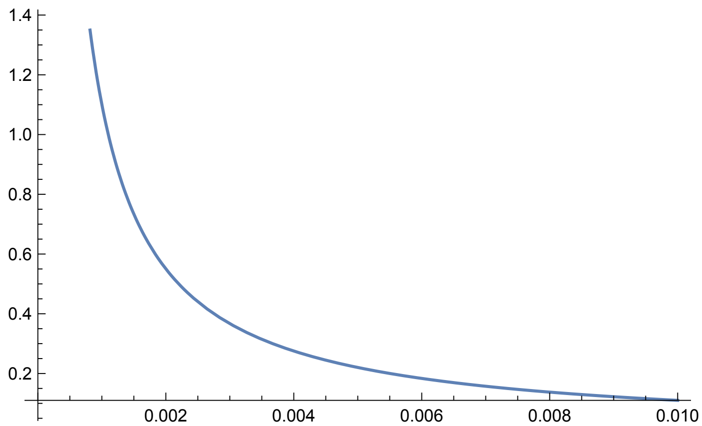
```
In[144]:= s = 0.006529828886969226
Out[144]= 0.00652983

In[145]:= edistNorm = NormalDistribution[-0.00011849766017034486`, 0.004097422554612836`]
Out[145]= NormalDistribution[-0.000118498, 0.00409742]

In[146]:= (* 10 m distribution for USDC-ETH *)

In[147]:= edistWethUsdc90dFiltered10MinCandle = StableDistribution[1, 1.4646547203677354`,
            -0.04962071457296463`, -0.000010553047315527044`, 0.0015427669619594313`]
Out[147]= StableDistribution[1, 1.46465, -0.0496207, -0.000010553, 0.00154277]

In[148]:= edistWethUsdc90dFiltered1hCandle = StableDistribution[1,
            1.4646547203677354`, -0.04962071457296463`, -0.000010553047315527044` * 6,
            0.0015427669619594313` * (6 / 1.4646547203677354`) ^ (1 / 1.4646547203677354`)]
Out[148]= StableDistribution[1, 1.46465, -0.0496207, -0.0000633183, 0.00404045]

In[149]:= (* Great *)
          (* Now be super rigorous w it! *)

In[150]:= CDF[StableDistribution[1, 1.46465, -0.0496207, -0.000010553, 0.00154277], 3]
Out[150]= 0.999997

In[151]:= InverseCDF[edistWethUsdc90dFiltered10MinCandle, 0.99]
Out[151]= 0.0124991

In[152]:= mu = -0.00011849766017034486`
Out[152]= -0.000118498

In[153]:= sig = 0.004097422554612836`
Out[153]= 0.00409742

In[154]:= CDF[NormalDistribution[0, 1], 0]
Out[154]= 1/2

In[155]:= CDF[NormalDistribution[0, 1], -sig]
Out[155]= 0.498365

In[156]:= (1 - CDF[NormalDistribution[0, 1], -sig]) / (1 - CDF[NormalDistribution[0, 1], 0])
Out[156]= 1.00327

In[157]:= Log[1.003269261047541`]
Out[157]= 0.00326393
```

```
In[158]:= 0.0032639286325196527` + sig^2 / 2.0
Out[158]= 0.00327232

In[159]:= (* Perfect .... this is > 0 so can use l * Q to make this go to 0 *)

In[160]:= mu
Out[160]= -0.000118498

In[161]:= (* Bring in mu *)

In[162]:= CDF[NormalDistribution[0, 1], -(mu - 2 * s) / sig - sig]
Out[162]= 0.999341

In[163]:= 1 - CDF[NormalDistribution[mu - 2 * s, sig], 0]
Out[163]= 0.000649488

In[164]:= CDF[NormalDistribution[mu - 2 * s, sig], 0]
Out[164]= 0.999351

In[165]:= CDF[NormalDistribution[0, 1], -(mu - 2 * s) / sig]
Out[165]= 0.999351

In[166]:= (1 - CDF[NormalDistribution[0, 1], -(mu - 2 * s) / sig - sig]) /
           (1 - CDF[NormalDistribution[0, 1], -(mu - 2 * s) / sig])
Out[166]= 1.01437

In[167]:= (* even better! Beautiful :) ... mu + sig^2/2 - l * Q + Log[(1-CDF)/(1-CDF)] ≤
           0 to have negative EV at Q *)

In[168]:= Exp[mu + sig^2 / 2.0]
Out[168]= 0.99989

In[169]:= mu + sig^2 / 2.0
Out[169]= -0.000110103

In[170]:= 2 * s
Out[170]= 0.0130597

In[171]:= 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[171]= 0.00110089

In[172]:= (* What's the expected PnL given > s when lambda = 0? *)
```

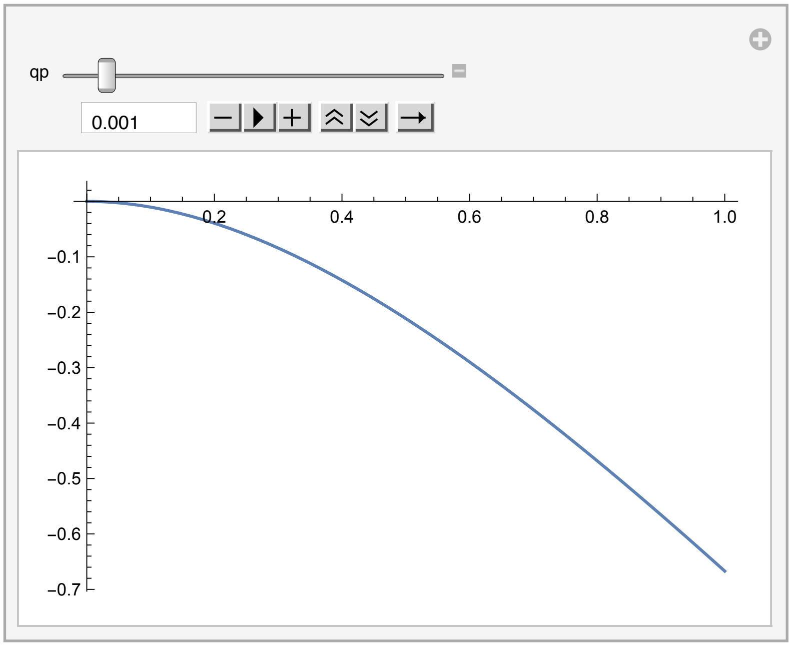
In[173]:=
```
Exp[rhs] - 1
```
Out[173]=
```
0.0011015
```
In[174]:=
```
(* 0.11% in 10m interval as scalp is pretty small already. Now,
ensure we make that EV ≤ 0 ... *)
```
In[175]:=
```
(* need l*Q ≥ rhs for negative EV. Take l'*(Q/Qmax) ≥ rhs ... l' ≥
 rhs * (Qmax/Q) *)
```
In[176]:=
```
(* if take l' = rhs * (Qmax/Q0), know that EV < 0 whenever Q > Q0 *)
```
In[177]:=
```
lNormUsdcWethPrime[q_] := (1 / q) * rhs
```
In[178]:=
```
Plot[lNormUsdcWethPrime[q], {q, 0, 0.01}]
```
Out[178]=



In[179]:=
```
Exp[1.5]
```
Out[179]=
```
4.48169
```
In[180]:=
```
Exp[-1.5]
```
Out[180]=
```
0.22313
```
In[181]:=
```
(* Check norm EV values given Q0 choice *)
```
In[182]:=
```
evNormUsdcWeth[qp_, q_] := q * (Exp[rhs - lNormUsdcWethPrime[qp] * q] - 1)
```

In[183]:=
```
Manipulate[Plot[evNormUsdcWeth[qp, q], {q, 0, 1}], {qp, 0.00025, 0.01}]
```

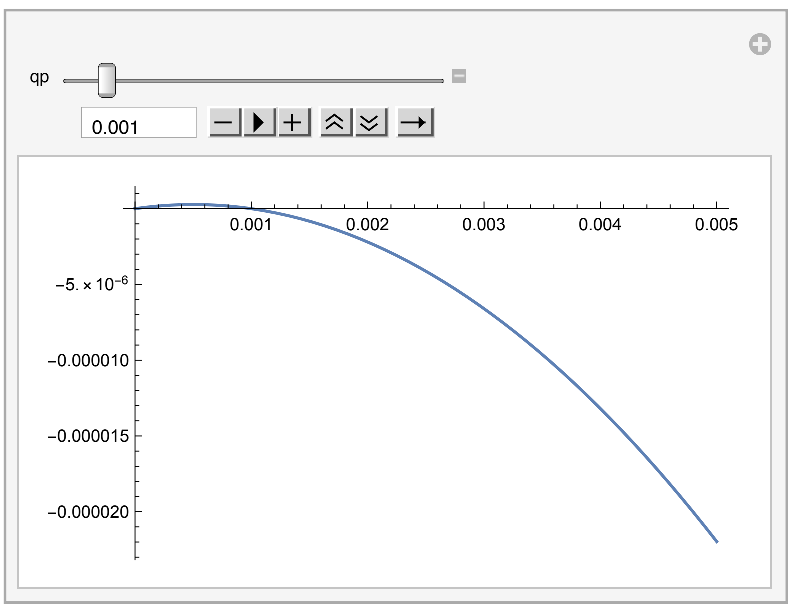
Out[183]=



In[184]:=
```
(* Zoom in around 1% mark ... *)
```

In[185]:=
```
Manipulate[Plot[evNormUsdcWeth[qp, q], {q, 0, 0.005}], {qp, 0.00025, 0.01}]
```

Out[185]=



In[186]:=
```
(* EV of < 0.01 bps of OI max/cap on scalp if take lprime to happen at Q =
 1% of cap *)
```

In[187]:=
```
(* Take 0.1% of cap as the l anchor *)
```

In[188]:=
```
lNormUsdcWethPrime[0.001]
```

Out[188]=
```
1.10089
```

In[189]:=
```
Exp[lNormUsdcWethPrime[0.001]]
```
Out[189]=
```
3.00685
```

In[190]:=
```
Exp[-lNormUsdcWethPrime[0.001]]
```
Out[190]=
```
0.332573
```

In[191]:=
```
(* if take up 100% of cap,
will have 4x slippage to upside and -76% slippage to downside *)
```

In[192]:=
```
(* What's the slippage function? Compare vs Uniswap as well ... *)
```

In[193]:=
```
(* Us: P/P0 - 1 = e^{s+l*q} - 1; Uniswap: P/P0 - 1 = (1+q)^2 - 1 *)
```

In[194]:=
```
(* take s = 0 for here *)
```

In[195]:=
```
slippageNormUsdcsWeth[qp_, q_] := Exp[lNormUsdcWethPrime[qp] * q] - 1
```

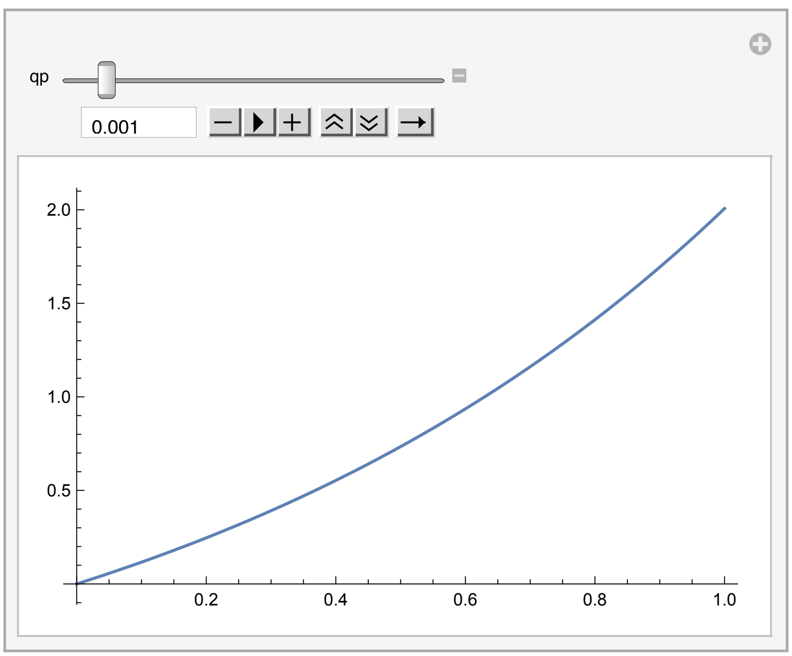
In[196]:=
```
slippageNormWithSpreadUsdcsWeth[qp_, q_] :=
 Exp[s * Sign[q] + lNormUsdcWethPrime[qp] * q] - 1
```

In[197]:=
```
uniswapSlippageUp[q_] := (1 + q) ^2 - 1
```

In[198]:=
```
uniswapSlippageDown[q_] := 1 / (1 + q) ^2 - 1
```

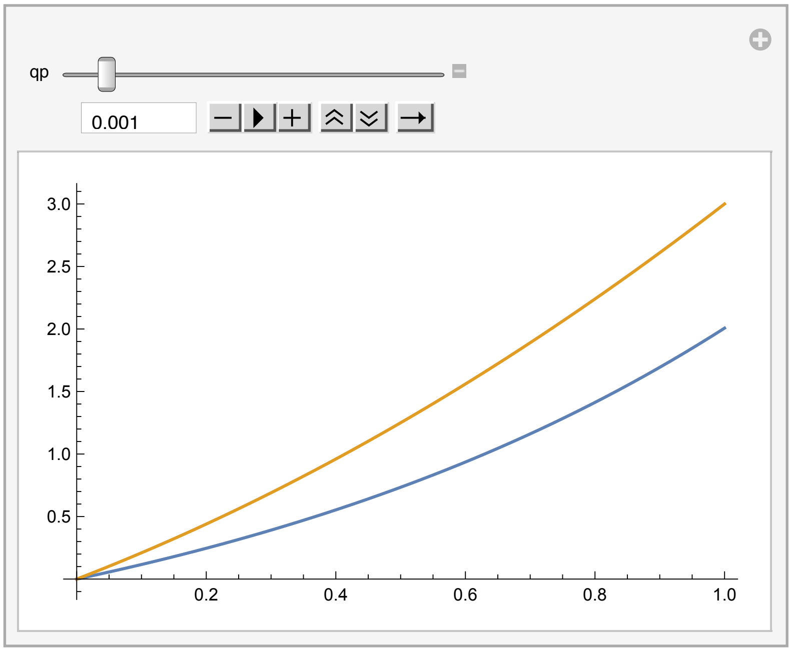
In[199]:=
```
Manipulate[Plot[slippageNormUsdcsWeth[qp, q], {q, 0, 1}, PlotRange → All],
 {qp, 0.00025, 0.01}]
```

Out[199]=



In[200]:=
```
Manipulate[Plot[{slippageNormUsdcsWeth[qp, x], uniswapSlippageUp[x] },
  {x, 0, 1.0}], {qp, 0.00025, 0.01}]
```

Out[200]=



In[201]:=
```
(* Values that matter for
 us: q in [0, 1] where q is percentage of OI cap user takes up *)
```

In[202]:=
```
(* For values that matter,
can calibrate slippage such that scalp trade is -EV for traders taking up >
 0.1% of cap when price jumps *)
```

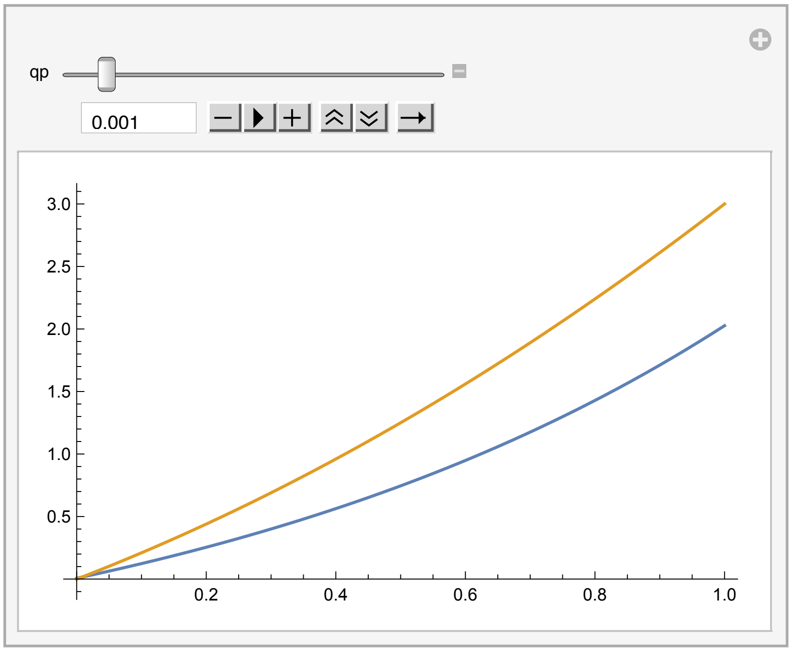
In[203]:=
```
(* We are the blue. Uniswap V2 x*y=
 k is the orange in the same region. We can have less slippage than
  Uniswap in all regions of interest. Chart above assumes our OI max is
  the same as 1/2 entire Uniswap liquidity (# of 'x' tokens in pool) *)
```

In[204]:=

In[205]:=
```
(* Compare with the static spread as well ... *)
```

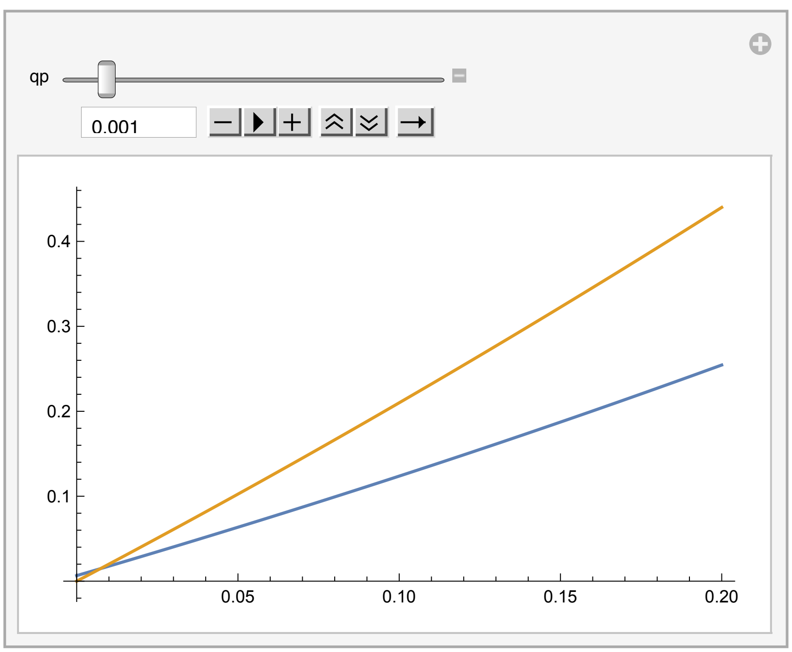
In[206]:=
```
Manipulate[Plot[{slippageNormWithSpreadUsdcsWeth[qp, x], uniswapSlippageUp[x] },
  {x, 0, 1.0}], {qp, 0.00025, 0.01}]
```

Out[206]=



In[207]:=
```
Manipulate[Plot[{slippageNormWithSpreadUsdcsWeth[qp, x], uniswapSlippageUp[x] },
  {x, 0, 0.2}], {qp, 0.00025, 0.01}]
```

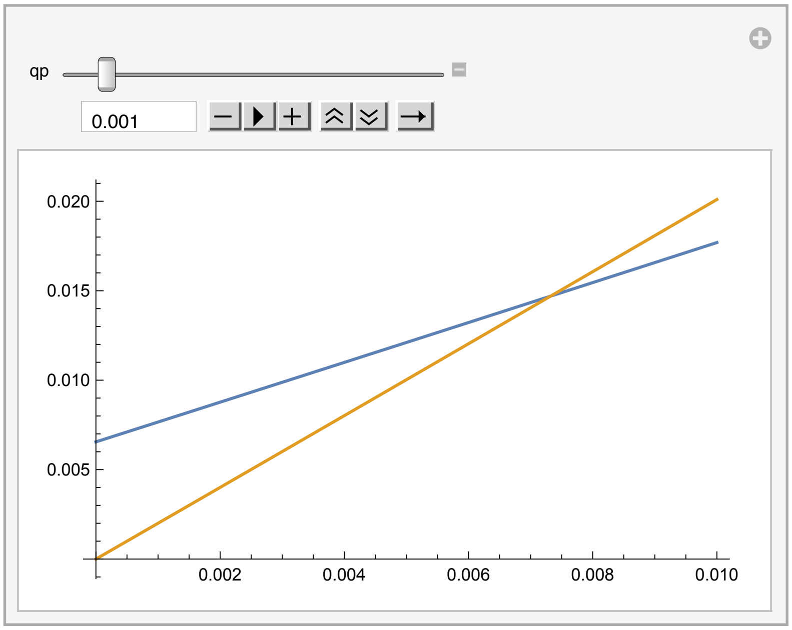
Out[207]=



In[208]:=
```
(* Still beautiful. Zoom in around x = 0 *)
```

In[209]:=
```
Manipulate[Plot[{slippageNormWithSpreadUsdcsWeth[qp, x], uniswapSlippageUp[x] },
  {x, 0, 0.01}], {qp, 0.00025, 0.01}]
```

Out[209]=

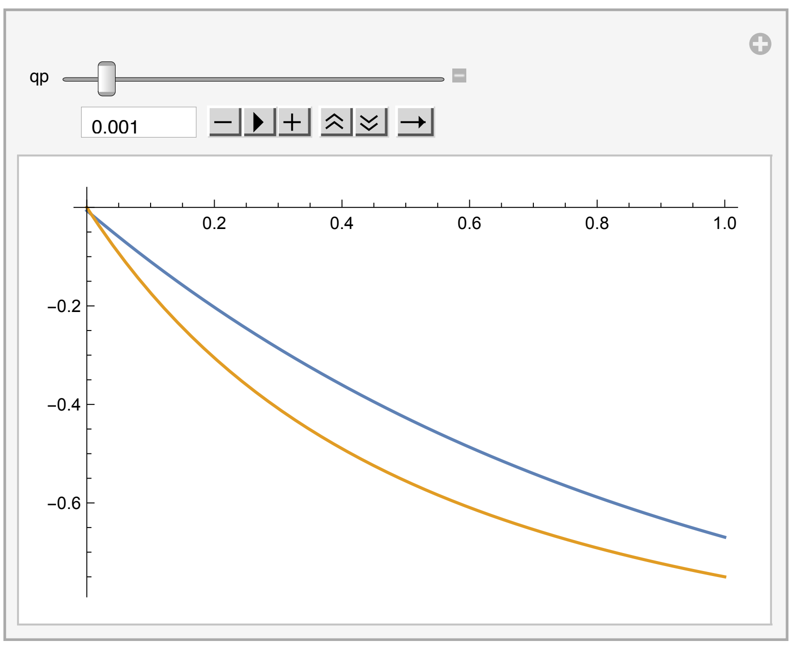


In[211]:=
```
(* And the static spread is apparent :) *)
```

In[212]:=
```
(* What about slippage on the downside? *)
```

In[213]:=
```
Manipulate[
 Plot[{slippageNormWithSpreadUsdcsWeth[qp, -x], uniswapSlippageDown[x] },
  {x, 0, 1.0}], {qp, 0.00025, 0.01}]
```

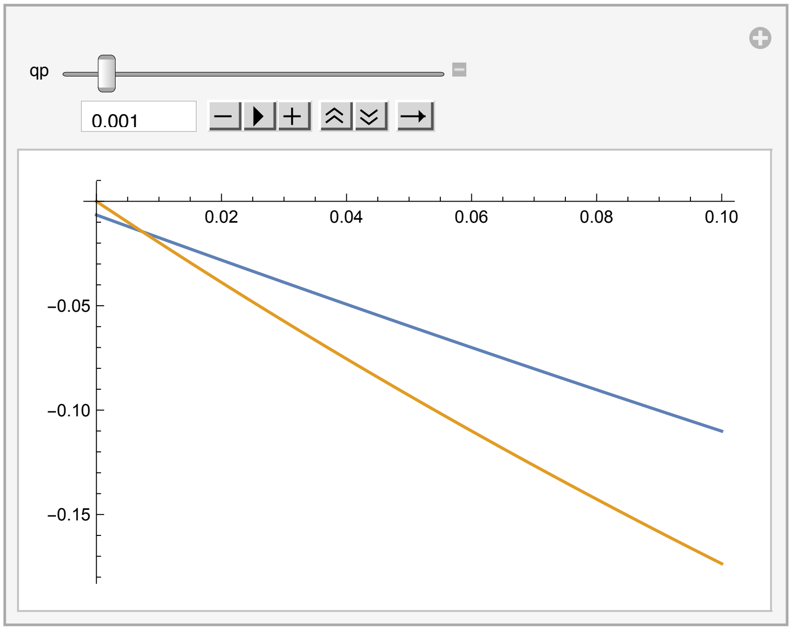
Out[213]=



In[214]:=
```
(* Beautiful here as well ... *)
```

In[215]:=
```
Manipulate[
 Plot[{slippageNormWithSpreadUsdcsWeth[qp, -x], uniswapSlippageDown[x] },
  {x, 0, 0.1}], {qp, 0.00025, 0.01}]
```

Out[215]=



In[216]:=
```
(* 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[217]:=
```
(* From Mathematica fit ... *)
```

In[218]:=
```
edistWethUsdc90dFiltered10MinCandle
```

Out[218]= `StableDistribution[1, 1.46465, -0.0496207, -0.000010553, 0.00154277]`

In[219]:=
```
cp = 4
```

Out[219]= 4

In[220]:=
```
Log[1.0 + cp]
```

Out[220]= 1.60944

In[221]:=
```
s
```

Out[221]= 0.00652983

In[222]:=
```
edistWethUsdcYv10MinCandle = StableDistribution[1, 1.4646547203677354`,
  -0.04962071457296463`, -0.000010553047315527044 - 2.0 * s, 0.0015427669619594313`]
```

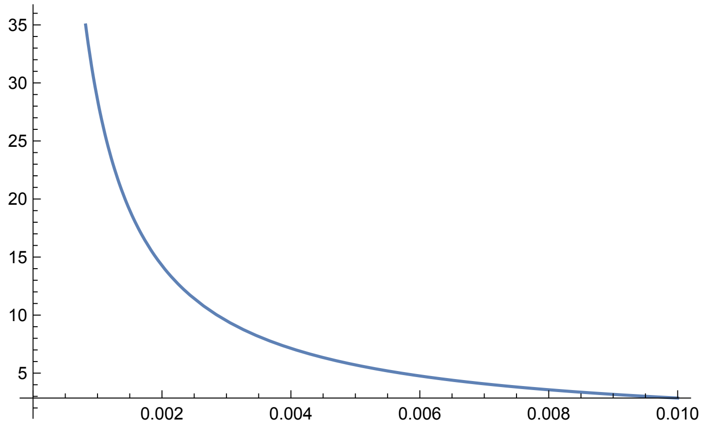
Out[222]= `StableDistribution[1, 1.46465, -0.0496207, -0.0130702, 0.00154277]`

In[223]:=
```
1 + cp
```

Out[223]= 5

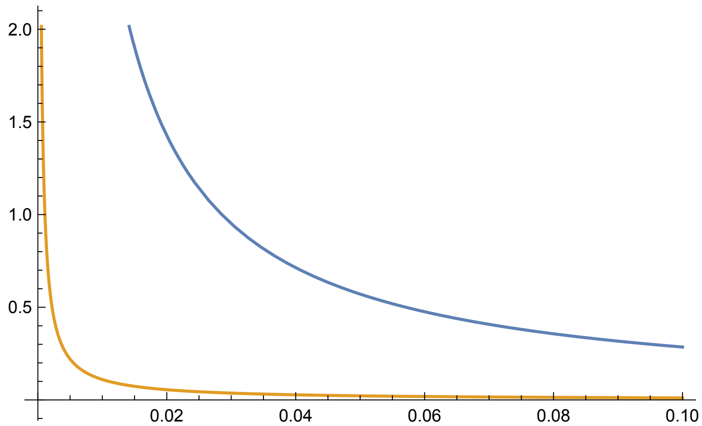
```
In[224]:= gInv = Log[1.0 + cp]

Out[224]= 1.60944

In[225]:= 1 - CDF[edistWethUsdcYv10MinCandle, 0]

Out[225]= 0.00933603

In[226]:= CDF[edistWethUsdcYv10MinCandle, 0]

Out[226]= 0.990664

In[227]:= (* Beautiful.... Less than normal *)

In[228]:= 1 - CDF[edistWethUsdcYv10MinCandle, gInv]

Out[228]= 7.48682 × 10^-6

In[229]:= CDF[edistWethUsdcYv10MinCandle, gInv]

Out[229]= 0.999993

In[230]:= (* This is to reach the 5x in 10m so negligible which is good *)

In[231]:= 1 - CDF[edistWethUsdcYv10MinCandle, 0] -
           (1 + cp) * (1 - CDF[edistWethUsdcYv10MinCandle, gInv])

Out[231]= 0.0092986

In[232]:= NIntegrate[PDF[edistWethUsdcYv10MinCandle, y] * Exp[y], {y, 0, gInv}]

Out[232]= 0.00956781

In[233]:= (NIntegrate[PDF[edistWethUsdcYv10MinCandle, y] * Exp[y], {y, 0, gInv}]) /
           (1 - CDF[edistWethUsdcYv10MinCandle, 0] -
             (1 + cp) * (1 - CDF[edistWethUsdcYv10MinCandle, gInv]))

Out[233]= 1.02895

In[234]:= logStableUsdcWeth =
           Log[(NIntegrate[PDF[edistWethUsdcYv10MinCandle, y] * Exp[y], {y, 0, gInv}]) /
             (1 - CDF[edistWethUsdcYv10MinCandle, 0] -
               (1 + cp) * (1 - CDF[edistWethUsdcYv10MinCandle, gInv]))]

Out[234]= 0.0285405

In[235]:= rhs

Out[235]= 0.00110089

In[236]:= (* way higher than EV from norm :)... 2.854% for stable vs 0.11% for norm *)

In[237]:= lStableUsdcWethPrime[q_] := (1 / q) * logStableUsdcWeth
```

In[238]:= `Plot[lStableUsdcWethPrime[q], {q, 0, 0.01}]`

Out[238]=


In[239]:= `(* Compare with normal ... *)`

In[240]:= `Plot[{lStableUsdcWethPrime[q], lNormUsdcWethPrime[q]}, {q, 0, 0.1}]`

Out[240]=


In[241]:= `lStableUsdcWethPrime[0.01]`

Out[241]= `2.85405`

In[242]:= `lNormUsdcWethPrime[0.01]`

Out[242]= `0.110089`

In[243]:= `lStableUsdcWethPrime[0.01] / lNormUsdcWethPrime[0.01] - 1`

Out[243]= `24.9248`

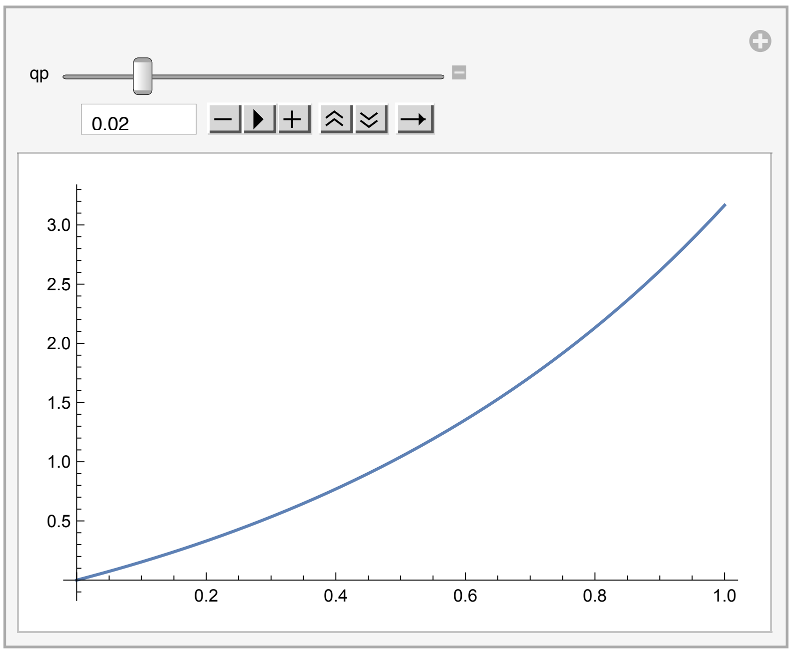
In[244]:=
```
(* 25x difference! in market impact
 parameter to obtain negative EV for 1% of OI cap *)
```

In[245]:= `(* 1% of OI for negative EV is likely what we want. Check slippage! *)`

In[246]:= `Exp[lNormUsdcWethPrime[0.01]]`

Out[246]= `1.11638`

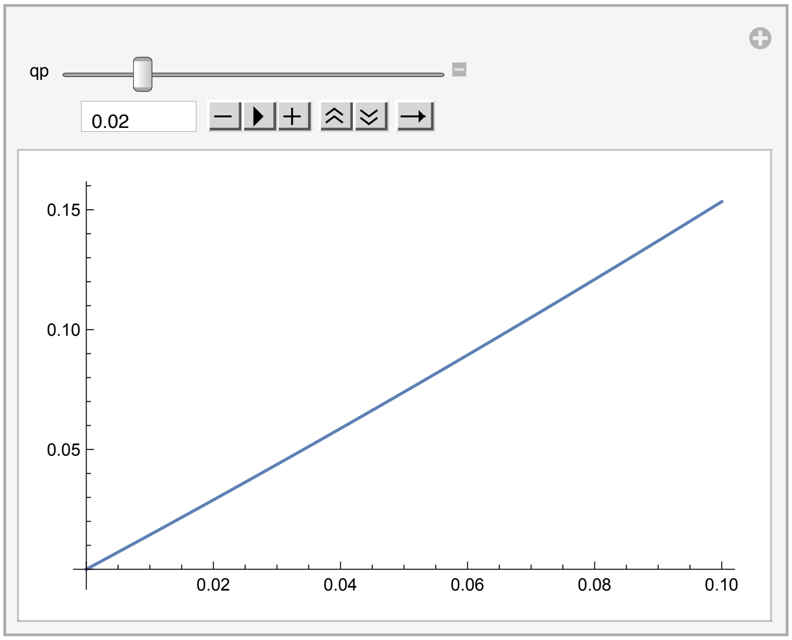
In[247]:=
```
Exp[lStableUsdcWethPrime[0.01]]
```
Out[247]=
```
17.3579
```
In[248]:=
```
Exp[-lStableUsdcWethPrime[0.01]]
```
Out[248]=
```
0.0576105
```
In[249]:=
```
slippageStableUsdcsWeth[qp_, q_] := Exp[lStableUsdcWethPrime[qp] * q] - 1
```
In[250]:=
```
slippageStableWithSpreadUsdcsWeth[qp_, q_] :=
 Exp[s * Sign[q] + lStableUsdcWethPrime[qp] * q] - 1
```
In[251]:=
```
Manipulate[Plot[slippageStableUsdcsWeth[qp, q], {q, 0, 1}, PlotRange → All],
 {qp, 0.0025, 0.1}]
```
Out[251]=



In[252]:=
```
(* Zoom in for smaller fish trades ... *)
```

In[253]:=
```
Manipulate[Plot[slippageStableUsdcsWeth[qp, q],
  {q, 0, 0.1}, PlotRange → All], {qp, 0.0025, 0.1}]
```

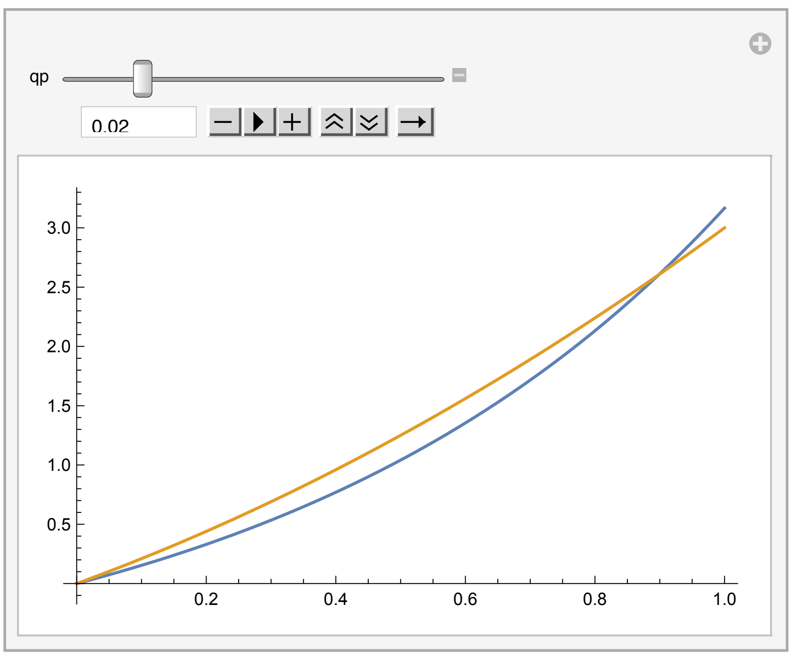
Out[253]=



In[254]:=
```
(* Compare w Uniswap ... *)
```

In[255]:=
```
Manipulate[Plot[{slippageStableUsdcsWeth[qp, x], uniswapSlippageUp[x] },
  {x, 0, 1.0}], {qp, 0.0025, 0.1}]
```

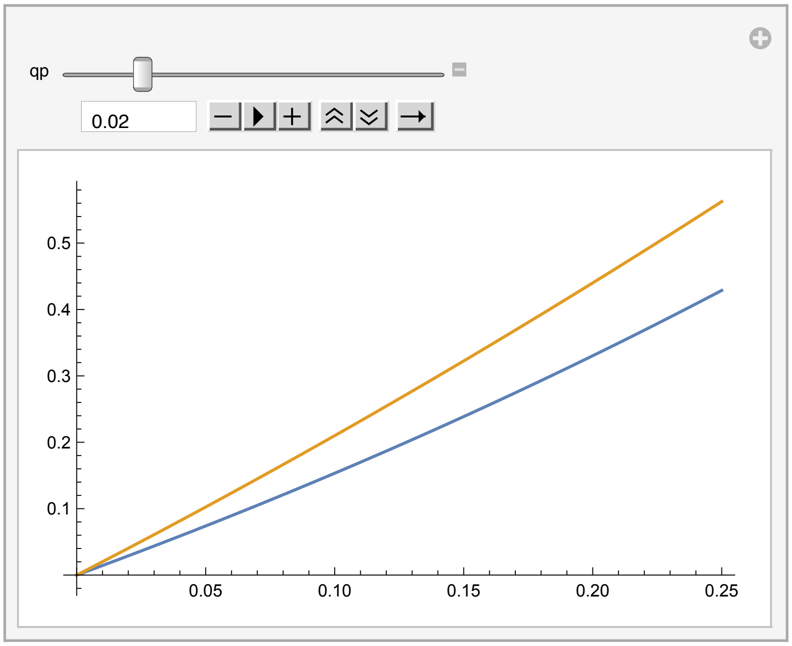
Out[255]=



In[256]:=
```
(* Zoom in for small fish *)
```

In[257]:=
```
Manipulate[Plot[{slippageStableUsdcsWeth[qp, x], uniswapSlippageUp[x] },
  {x, 0, 0.25}], {qp, 0.0025, 0.1}]
```

Out[257]=

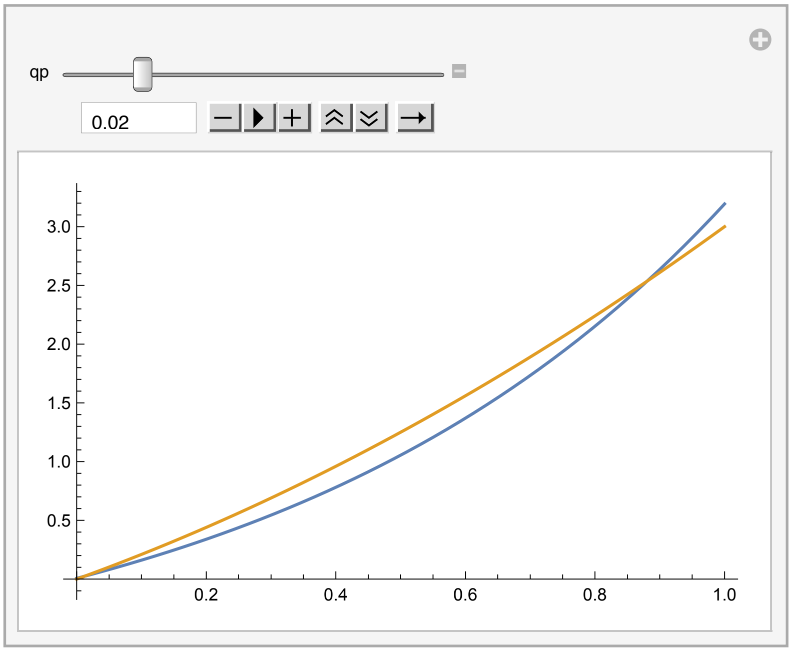


In[258]:=
```
(* If we risk 2% of the OI cap vs 1%,
we're at Uniswap slippage levels. This is beautiful *)
```

In[259]:=
```
(* Plot with static spread then check how much we can get milked for <=
    2% (worst case) ... max of EV given Q0 = 2% of Qmax .. *)
```

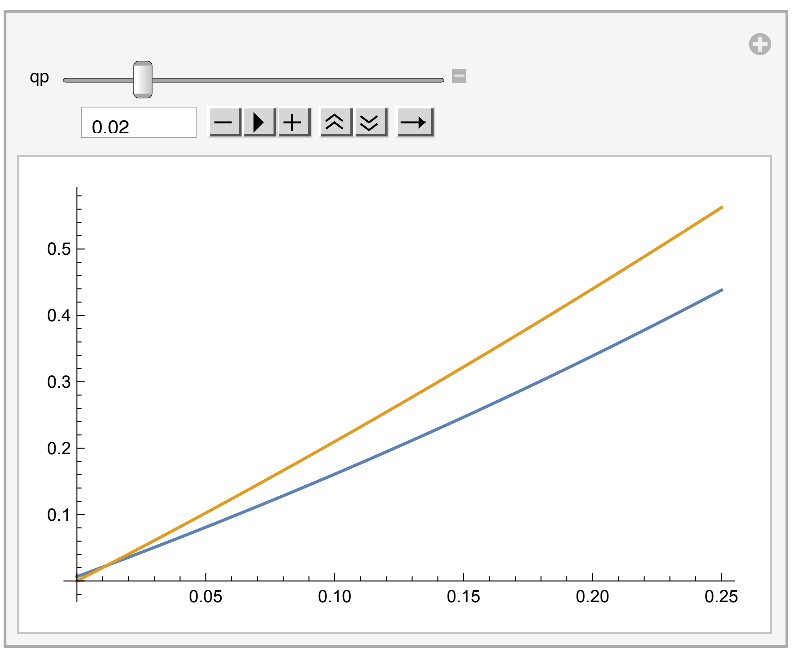
In[260]:=
```
Manipulate[Plot[{slippageStableWithSpreadUsdcsWeth[qp, x], uniswapSlippageUp[x] },
  {x, 0, 1.0}], {qp, 0.0025, 0.1}]
```

Out[260]=



In[261]:=
```
Manipulate[Plot[{slippageStableWithSpreadUsdcsWeth[qp, x], uniswapSlippageUp[x] },
  {x, 0, 0.25}], {qp, 0.0025, 0.1}]
```

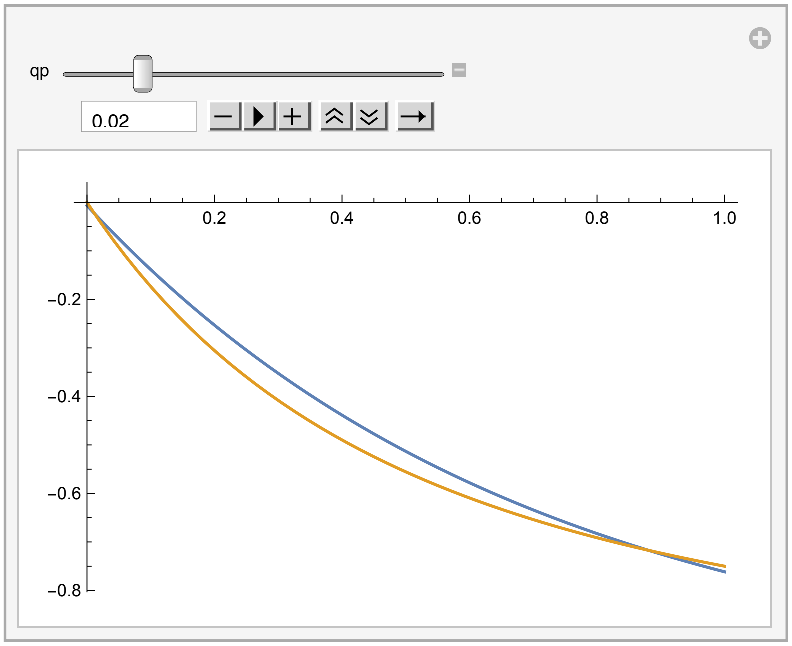
Out[261]=



In[262]:=
```
(* And on the downside ... *)
```

In[263]:=
```
Manipulate[
 Plot[{slippageStableWithSpreadUsdcsWeth[qp, -x], uniswapSlippageDown[x] },
  {x, 0, 1.0}], {qp, 0.0025, 0.1}]
```

Out[263]=



In[264]:=
```
h = Log[NIntegrate[PDF[edistWethUsdcYv10MinCandle, y] * Exp[y], {y, 0, gInv}] /
    (1 - CDF[edistWethUsdcYv10MinCandle, 0])]
```

Out[264]= 0.0245228

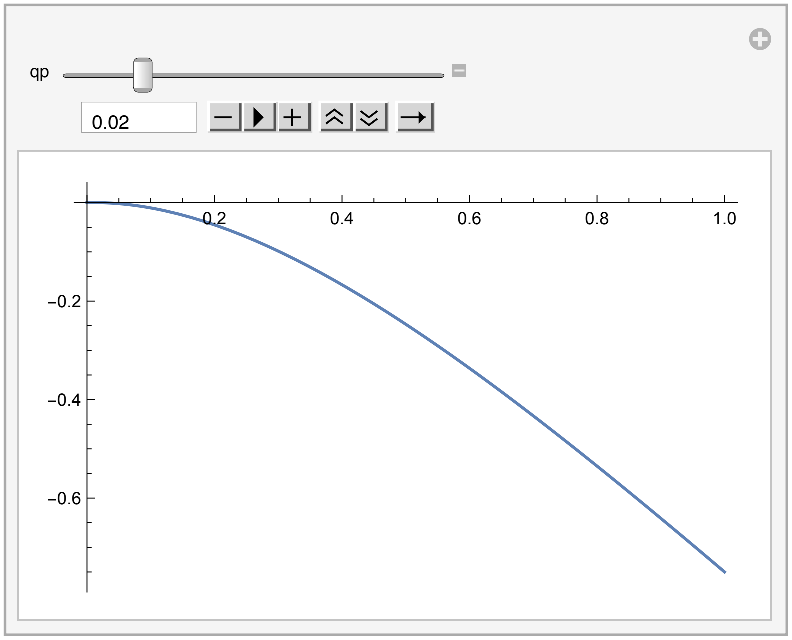
In[265]:=
```
rho = (1 + cp) * (1 - CDF[edistWethUsdcYv10MinCandle, gInv]) /
    (1 - CDF[edistWethUsdcYv10MinCandle, 0])
```

Out[265]= 0.00400964

In[266]:=
```
evStableUsdcWeth[qp_, q_] := q * (Exp[h - lStableUsdcWethPrime[qp] * q] - 1 + rho)
```

In[267]:=
```
Manipulate[Plot[evStableUsdcWeth[qp, q], {q, 0, 1}], {qp, 0.0025, 0.1}]
```

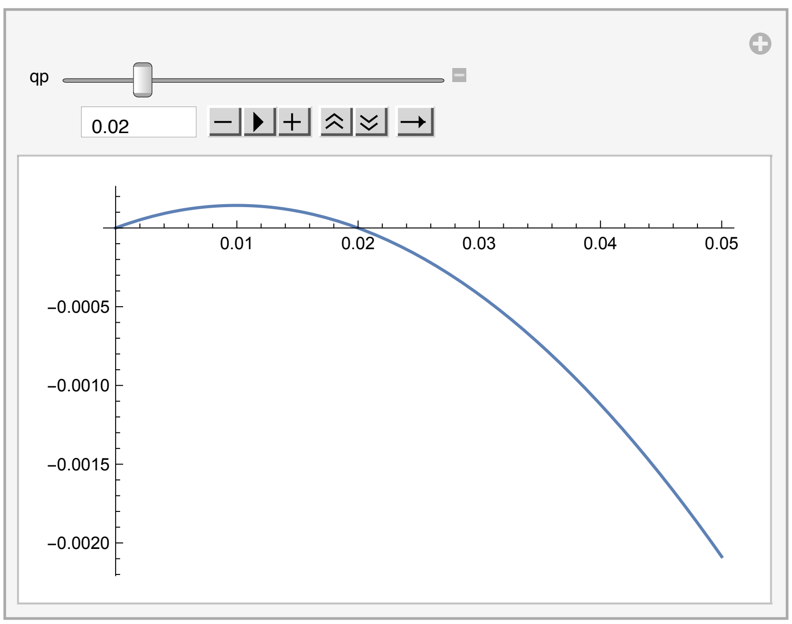
Out[267]=



In[268]:=
```
(* Zoom in where it really counts ... *)
```

In[269]:=
```
Manipulate[Plot[evStableUsdcWeth[qp, q], {q, 0, 0.05}], {qp, 0.0025, 0.1}]
```

Out[269]=



In[270]:=
```
(* 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[271]:=
```
evStableUsdcWeth[0.02, 0.01]
```

Out[271]=
```
0.000143149
```

In[272]:=
```
(* 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[273]:=
```
0.000143 * 100 000 000
```

Out[273]=
```
14 300.
```

In[274]:=
```
(* Since fees are about 15bps on each
  side => so 30 bps total. Eats into ~20% of scalps profit ... *)
```

In[275]:=
```
0.0030 * 1 000 000
```

Out[275]=
```
3000.
```

In[276]:=
```
3000 / 14 300.0
```

Out[276]=
```
0.20979
```

In[277]:=
```
(* How much more volume does it take to overcome that scalp if charging 30bps? *)
```

In[278]:=
```
14 300 - 3000.0
```

Out[278]=
```
11 300.
```

In[279]:=
```
11 300 / (0.0030)
```

Out[279]= 3.76667×10^{6}

In[280]:=
```
(* $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[281]:=
```
(* 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 *)
```