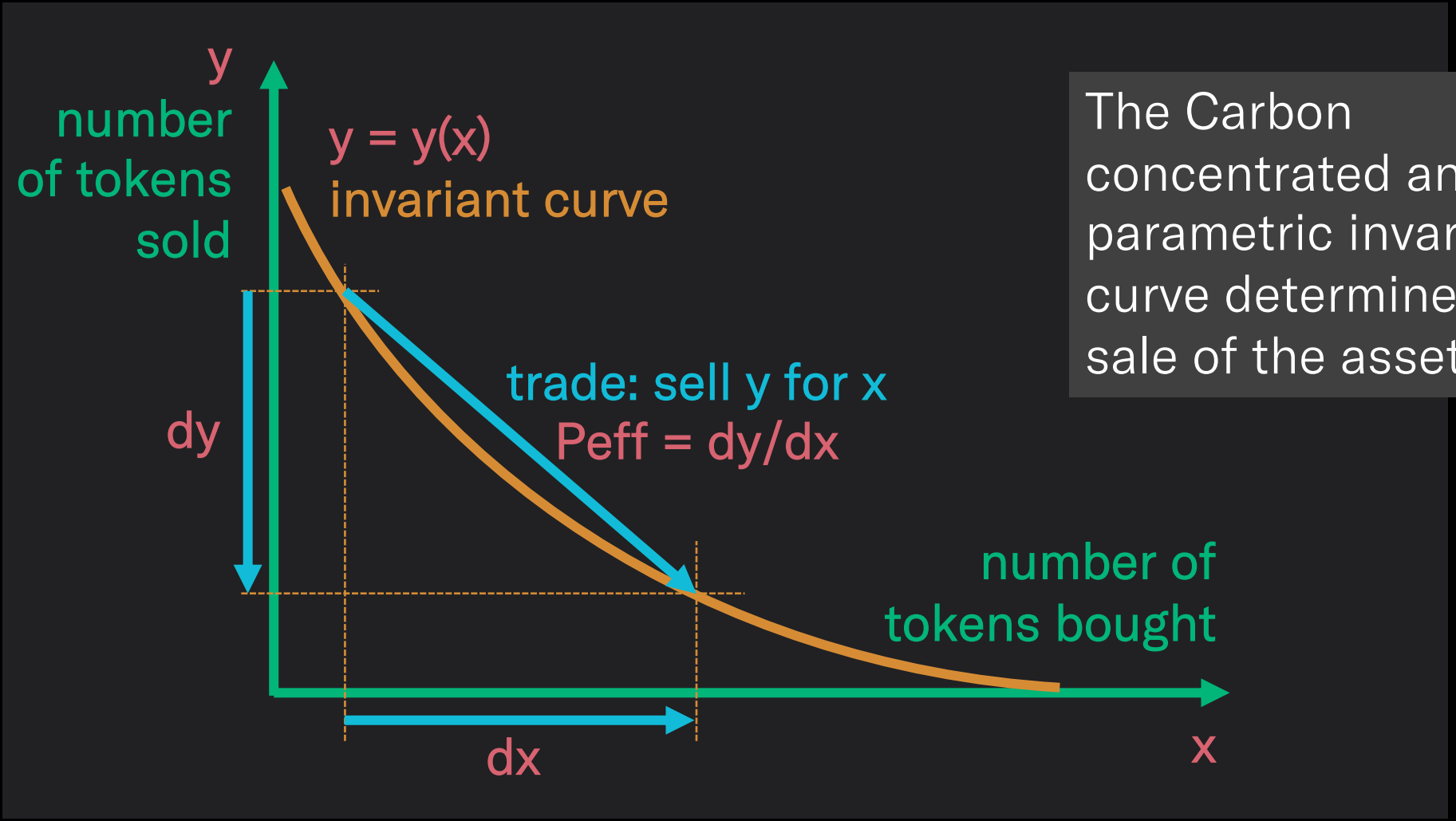
CARBON

Simulating Carbon

January 2023

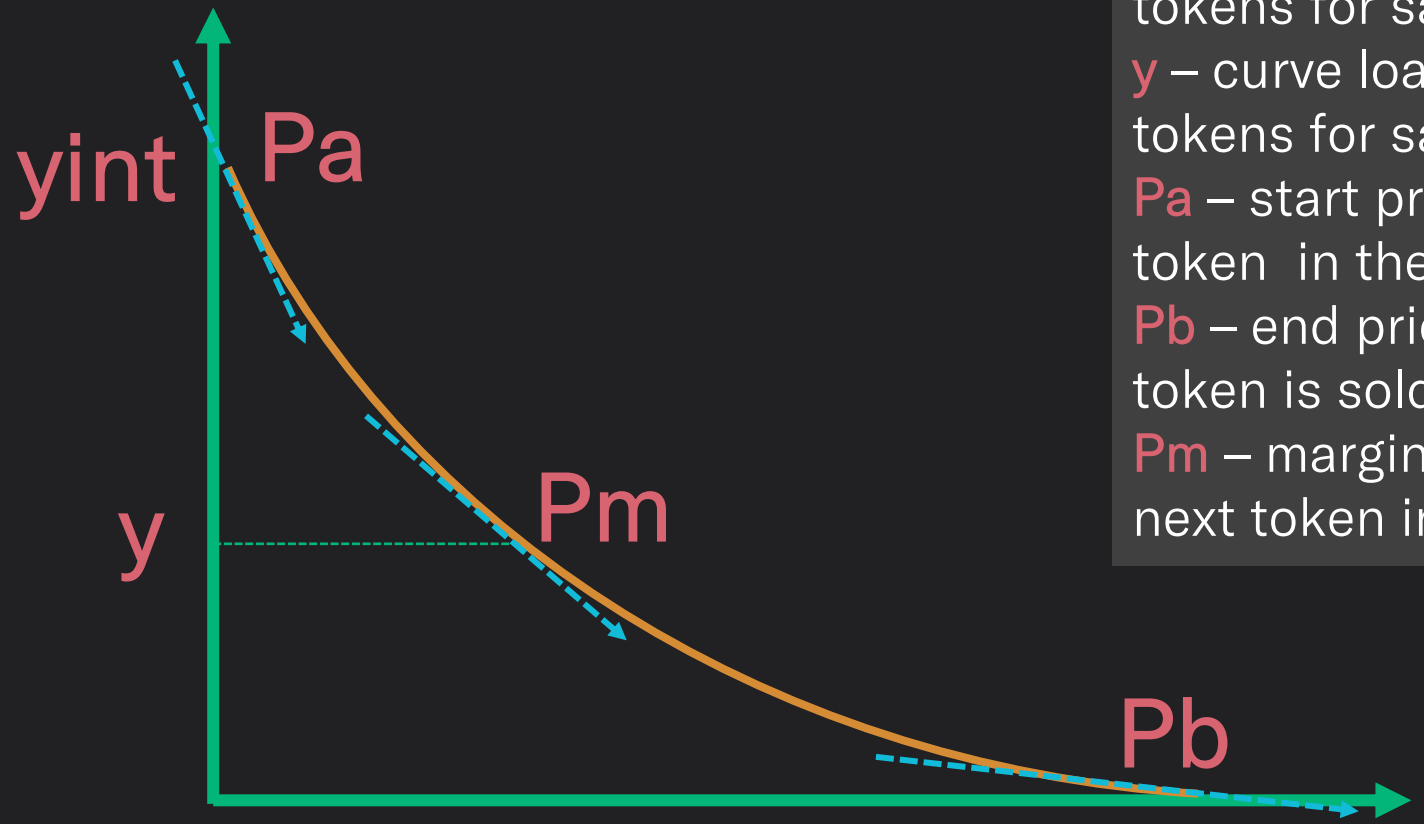
Carbon basics

The Carbon invariant curve



The Carbon concentrated and parametric invariant curve determines the sale of the asset y

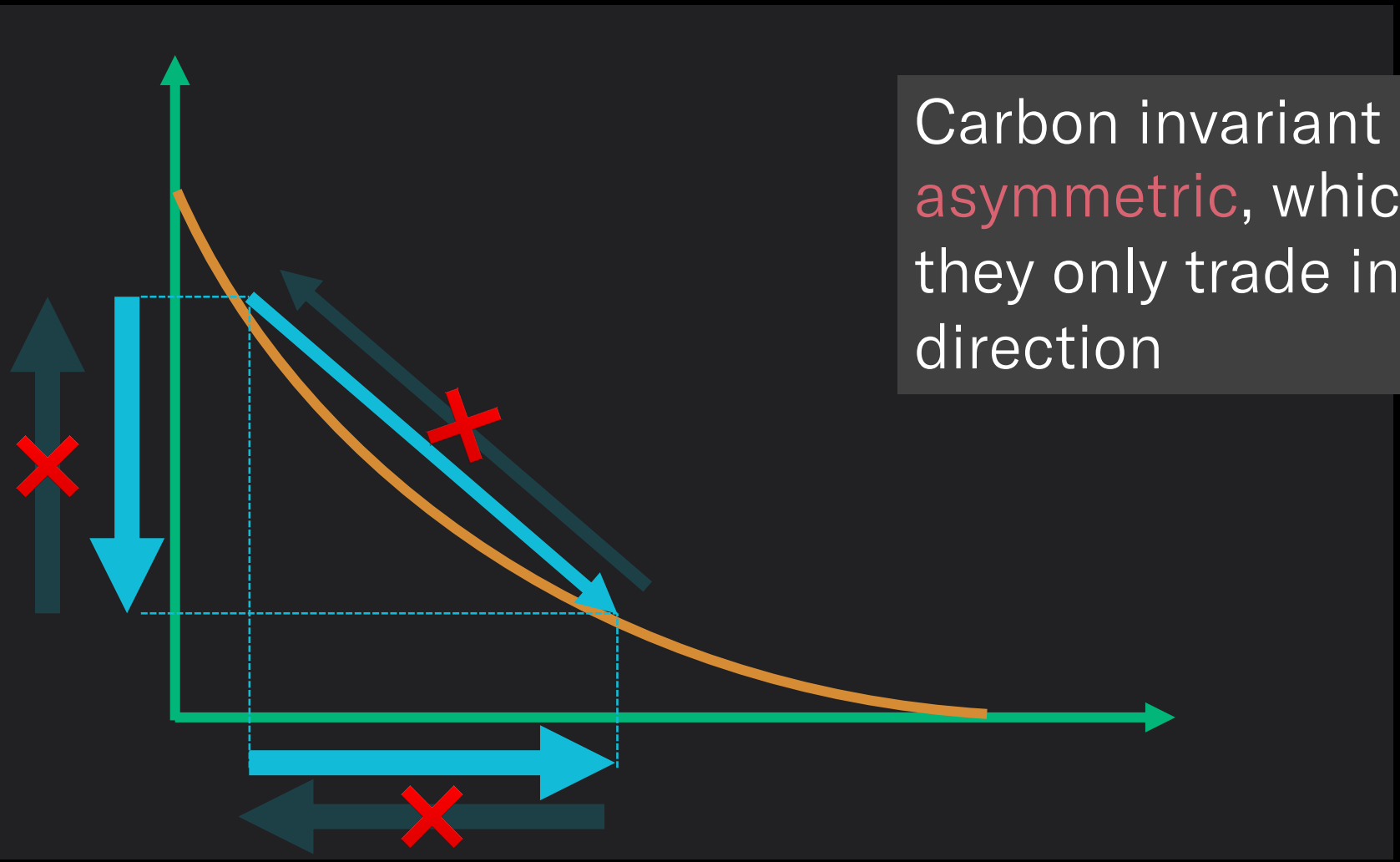
Carbon curve parameters



yint – curve capacity; maximum number of tokens for sale it can hold
y – curve loading; the current number of tokens for sale it holds
Pa – start price; the price at which the first token in the range is sold
Pb – end price; the price at which the last token is sold
Pm – marginal price; the price at which the next token in the range is sold

(1) Only four of five params independent.
(2) Natural price quote dy/dx, to be converted to pair convention

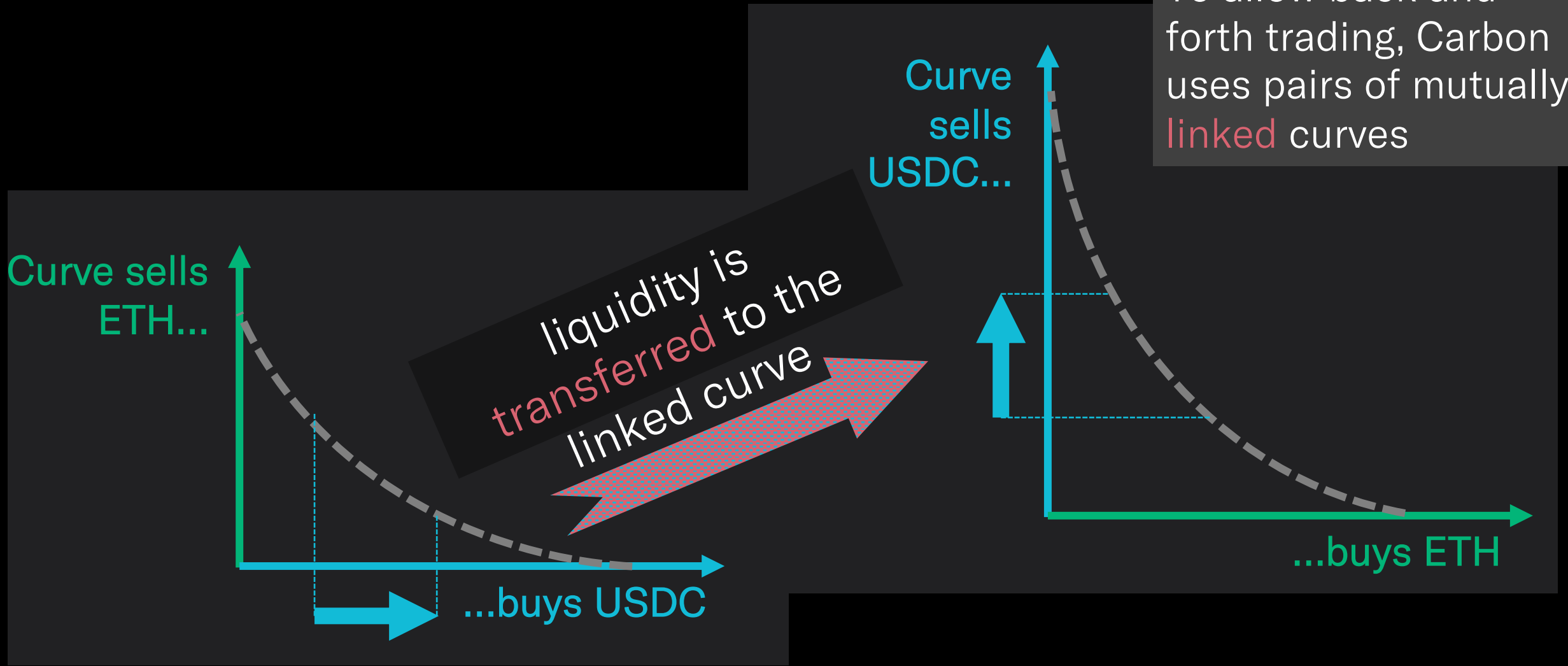
Carbon asymmetric curve



Carbon invariant curves are asymmetric, which means they only trade in one direction

Carbon linked curves

To allow back and forth trading, Carbon uses pairs of mutually linked curves



Minimum viable simulation

Minimum viable simulation

Initialize the simulator

```
Sim = CarbonSimulatorUI(pair="ETH/USDC")
```

Use ETH/USDC as default pair

Add a Carbon strategy

```
Sim.add_strategy(“ETH”,
                        1, 1500, 2000,
                    1000, 1250, 1000)
```

Sell ETH between 1,500-2,000; seed with 1 ETH
Buy ETH between 1,250-1,000; see with 1,000 USDC

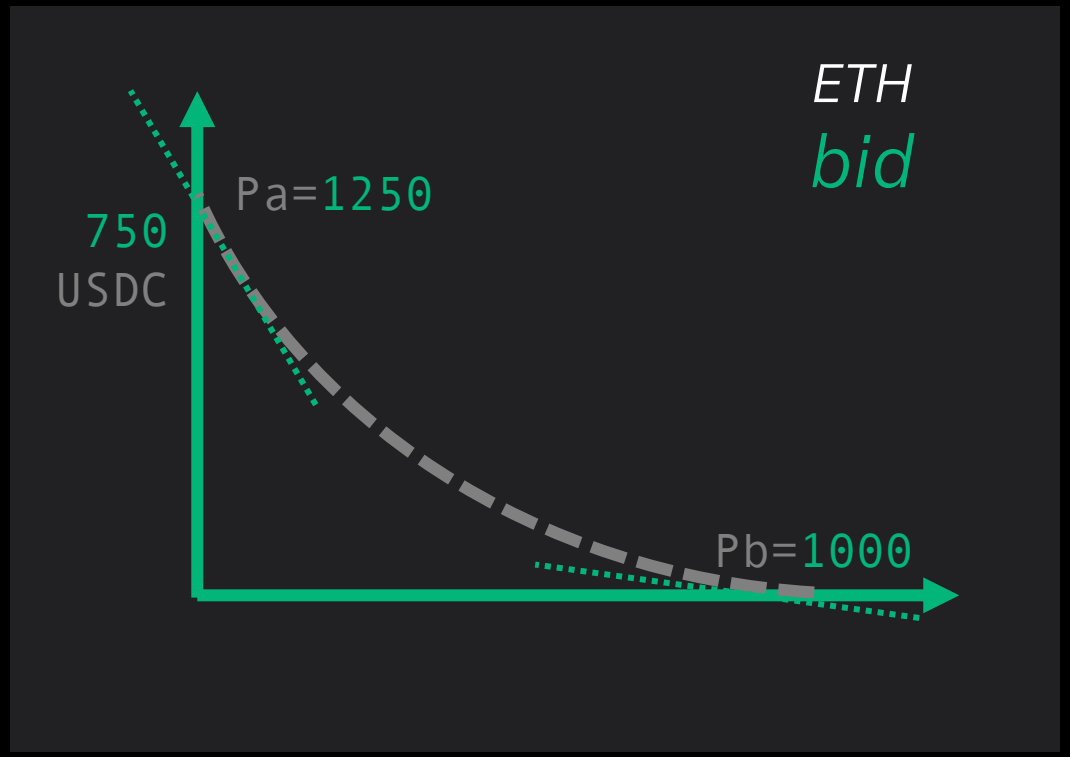
Execute a trade

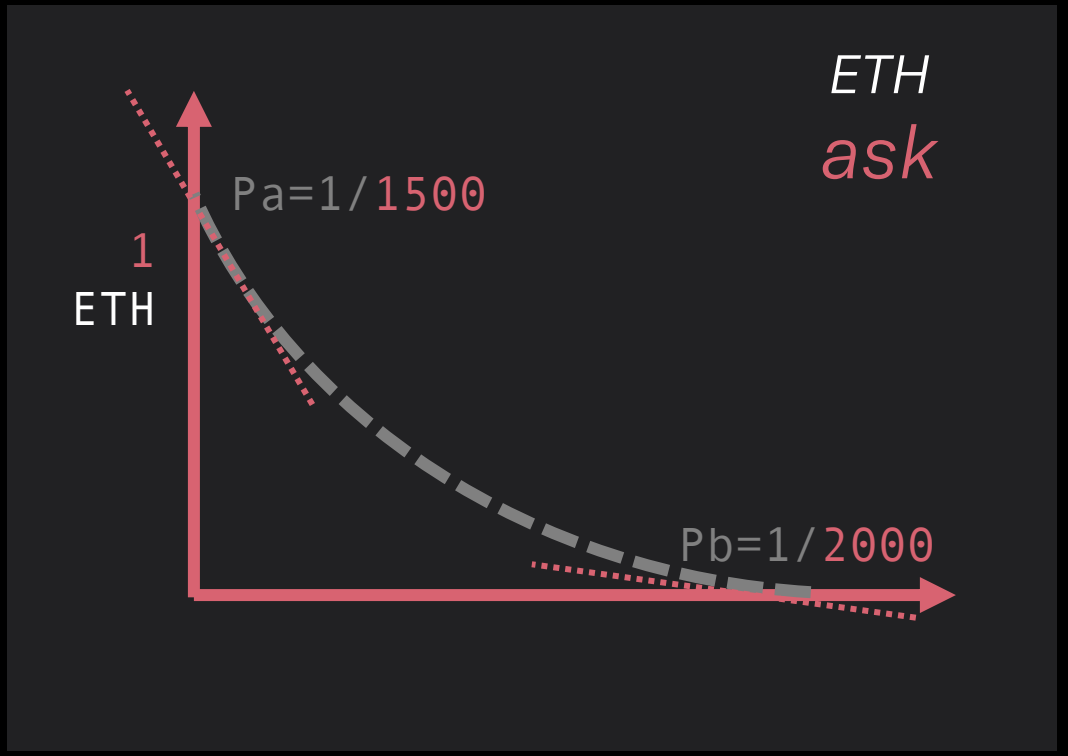
```
Sim.amm_sells(“ETH”, 0.5)
```

Sell 0.5 ETH

Adding a Carbon strategy

```
add_strategy("ETH", 1, 1500, 2000, 750, 1250, 1000)
```

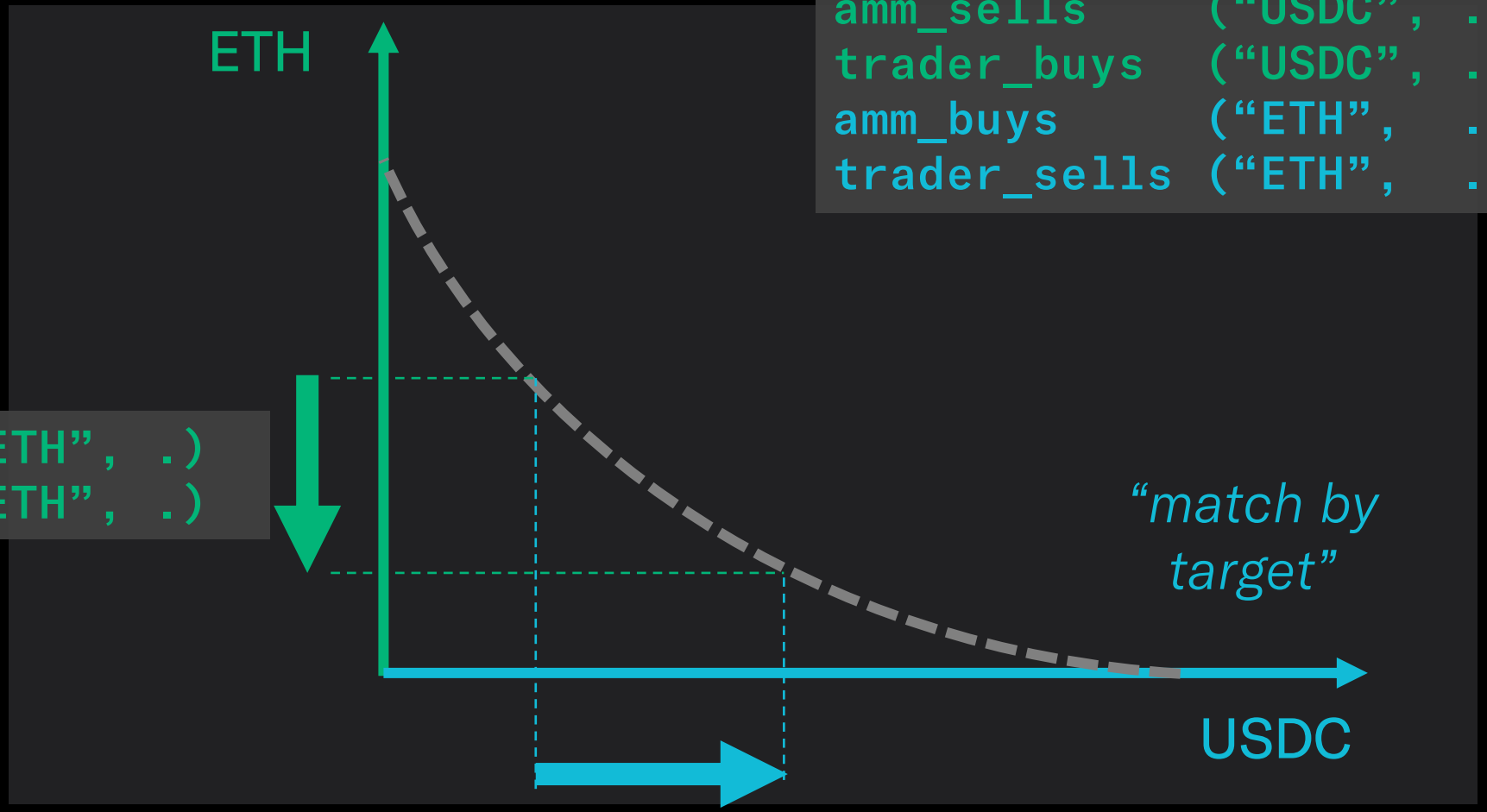




Executing a trade

```
LINKED CURVE
============
amm_sells     (“USDC”, .)
trader_buys   (“USDC”, .)
amm_buys      (“ETH”,  .)
trader_sells  (“ETH”,  .)
```

“match by source”




```
amm_sells    (“ETH”, .)
trader_buys  (“ETH”, .)
```

```
amm_buys      (“USDC”, .)
trader_sells  (“USDC”, .)
```

Simulation example

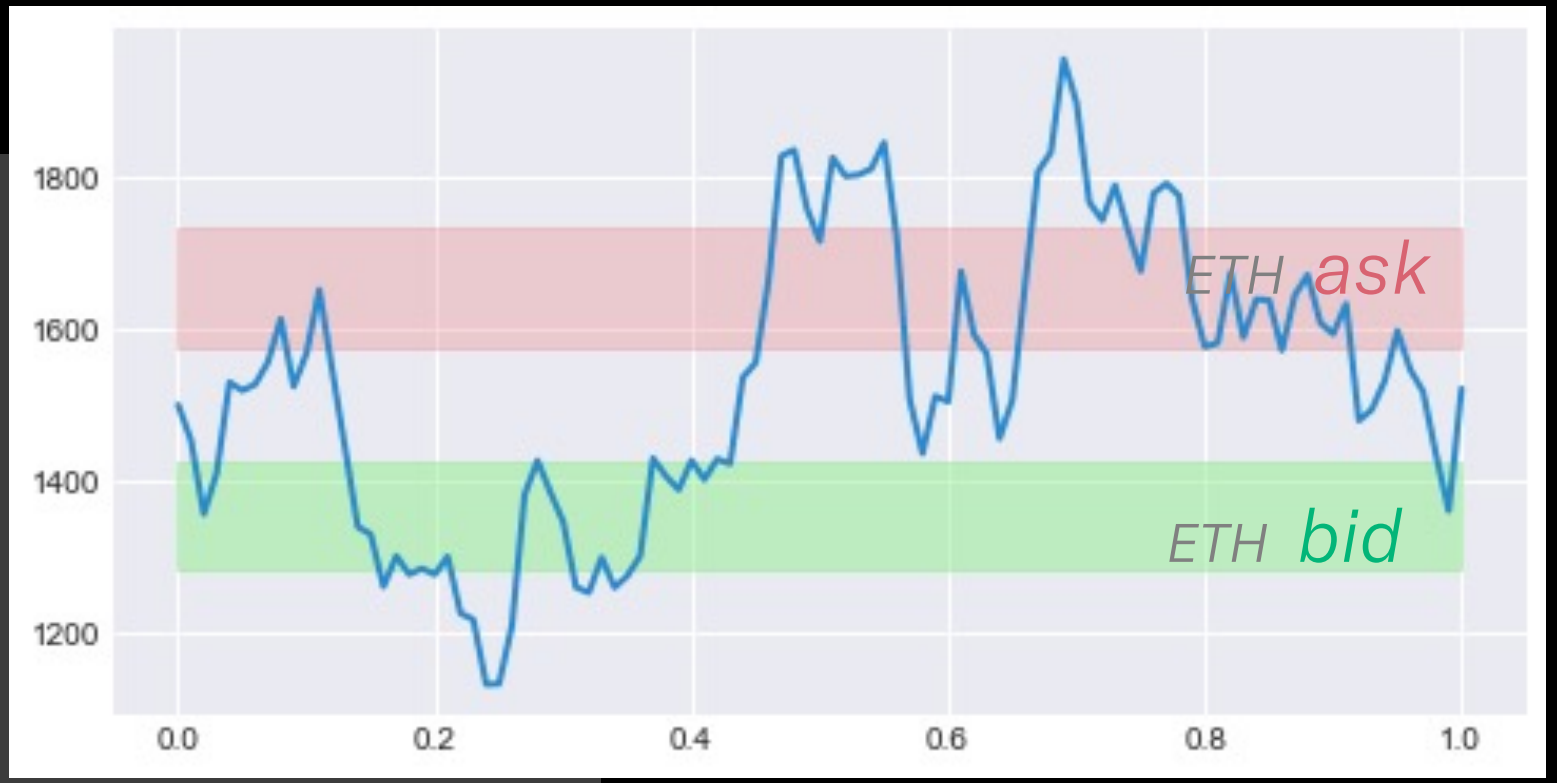
detailed sim using dyfromp_f function (Demo 7-1)

Carbon simulation run

Demo 7-1

```
    ETH bid 1282-1425 (750 USDC)
              spot 1500
    ETH ask 1575-1732 (  1 ETH )
```

```
buy ETH 1425.0-1282.5, sell ETH 1575.0-1732.5
--------------------
...
ix=   2, spot=1356.7: sell     472.93 USDC
...
ix=   8, spot=1612.9: sell       0.34 ETH
...
ix=  11, spot=1651.3: sell       0.33 ETH
...
ix=  14, spot=1340.0: sell     950.72 USDC
ix=  15, spot=1330.4: sell     109.97 USDC
ix=  16, spot=1261.5: sell     551.29 USDC
...
ix=  46, spot=1663.8: sell       1.08 ETH
ix=  47, spot=1826.8: sell       0.78 ETH
...
ix=  99, spot=1361.2: sell    1356.49 USDC
...
--------------------
ix=   0,  spot=1500.0: 1.0 ETH 1000.0 USDC (=2500.0 USDC)
ix= 100,  spot=1521.4: 1.0 ETH 1715.7 USDC (=3197.5 USDC)
```



Carbon simulation run – code overview

Demo 7-1

Simulation

```
]:
Sim = CarbonSimulatorUI(pair="ETH/USDC", verbose=False)
Sim.add_strategy("ETH", amt_eth, p_sell_a, p_sell_b, amt_usdc, p_buy_a, p_buy_b)
print()
print(f"buy ETH {p_buy_a:.1f}-{p_buy_b:.1f}, sell ETH {p_sell_a:.1f}-{p_sell_b:.1f}")
print("-"*20)
printdots = True
#Sim.state()["orders"]

for t, spot, ix in zip(time_r, path, range(len(path))):
    orderuis = Sim.state()["orderuis"]
    orders_sell_eth = {k:v for k,v in orderuis.items() if v.tkn=="ETH"}
    dy_f_sell_eth = lambda p: sum(o.dyfromp_f(p) for o in orders_sell_eth.values())
    sell_eth  = dy_f_sell_eth(spot)
    orders_sell_usdc = {k:v for k,v in orderuis.items() if v.tkn=="USDC"}
    dy_f_sell_usdc = lambda p: sum(o.dyfromp_f(p) for o in orders_sell_usdc.values())
    sell_usdc = dy_f_sell_usdc(spot)

    if sell_eth > 0.0001:
        r = Sim.amm_sells("ETH", sell_eth, support_partial=True)
        failed = "" if r['success'] else "FAILED"
        print(f"ix={ix:4.0f}, spot={spot:0.1f}: sell {sell_eth:10.2f} ETH {failed}")
        printdots = True

    elif sell_usdc > 0.001:
        r = Sim.amm_sells("USDC", sell_usdc, support_partial=True)
        failed = "" if r['success'] else "FAILED"
        print(f"ix={ix:4.0f}, spot={spot:0.1f}: sell {sell_usdc:10.2f} USDC {failed}")
        printdots = True

    else:
        if printdots:
            print("...")
        printdots = False
        #print(f"ix={ix:4.0f}, spot={spot:0.1f}: ---")

```

Initialize the simulator and load the strategy; then loop over the spot values

In the loop, determine how much ETH (and USDC) to sell to get to a certain price...

...and execute the transaction

Carbon simulation run – core code

Initialize the simulator and load the strategy; then loop over the spot values

Demo 7-1

Simulation

```
1  Sim = CarbonSimulatorUI(pair="ETH/USDC", verbose=False)
2  Sim.add_strategy("ETH", amt_eth, p_sell_a, p_sell_b, amt_usdc, p_buy_a, p_buy_b)
3  print()
```

In the loop, determine how much ETH (and USDC) to sell to get to a certain price…

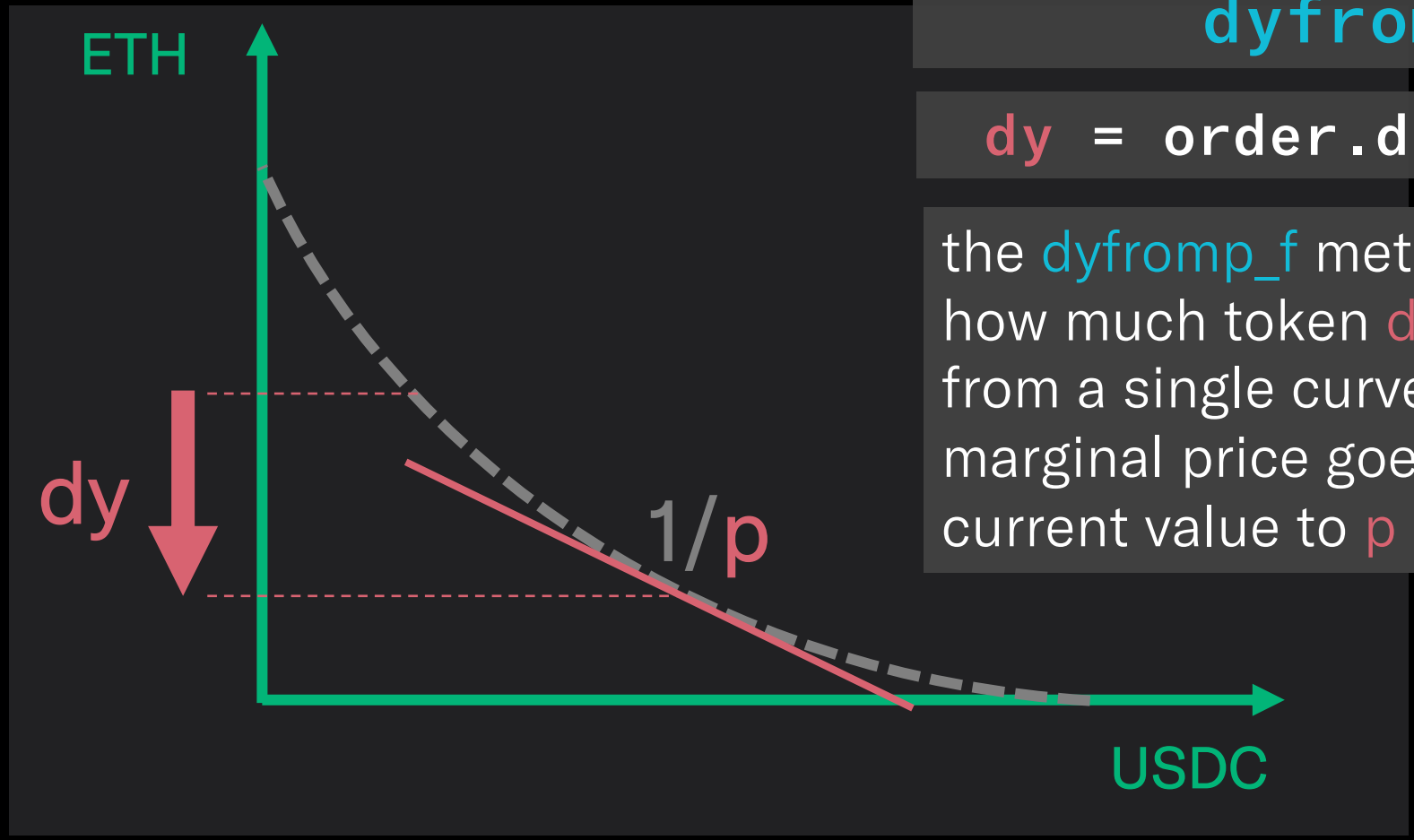
```
9  for t, spot, ix in zip(time_r, path, range(len(path))):
10     orderuis = Sim.state()["orderuis"]
11     orders_sell_eth = {k:v for k,v in orderuis.items() if v.tkn=="ETH"}
12     dy_f_sell_eth = lambda p: sum(o.dyfromp_f(p) for o in orders_sell_eth.values())
13     sell_eth  = dy_f_sell_eth(spot)
14     orders_sell_usdc = {k:v for k,v in orderuis.items() if v.tkn=="USDC"}
```

…and execute the transaction

```
18
19     if sell_eth > 0.0001:
20         r = Sim.amm_sells("ETH", sell_eth, support_partial=True)
21         failed = "" if r['success'] else "FAILED"
22         print(f"ix={ix:4.0f}, spot={spot:0.1f}: sell {sell_eth:10.2f
23         printdots = True
24
```

Liquidity released at a price

Demo 7-1



dyfromp_f

```
dy = order.dyfromp_f(p)
```

the dyfromp_f method calculates how much token dy will be sold from a single curve when the marginal price goes from its current value to p

Simulation example

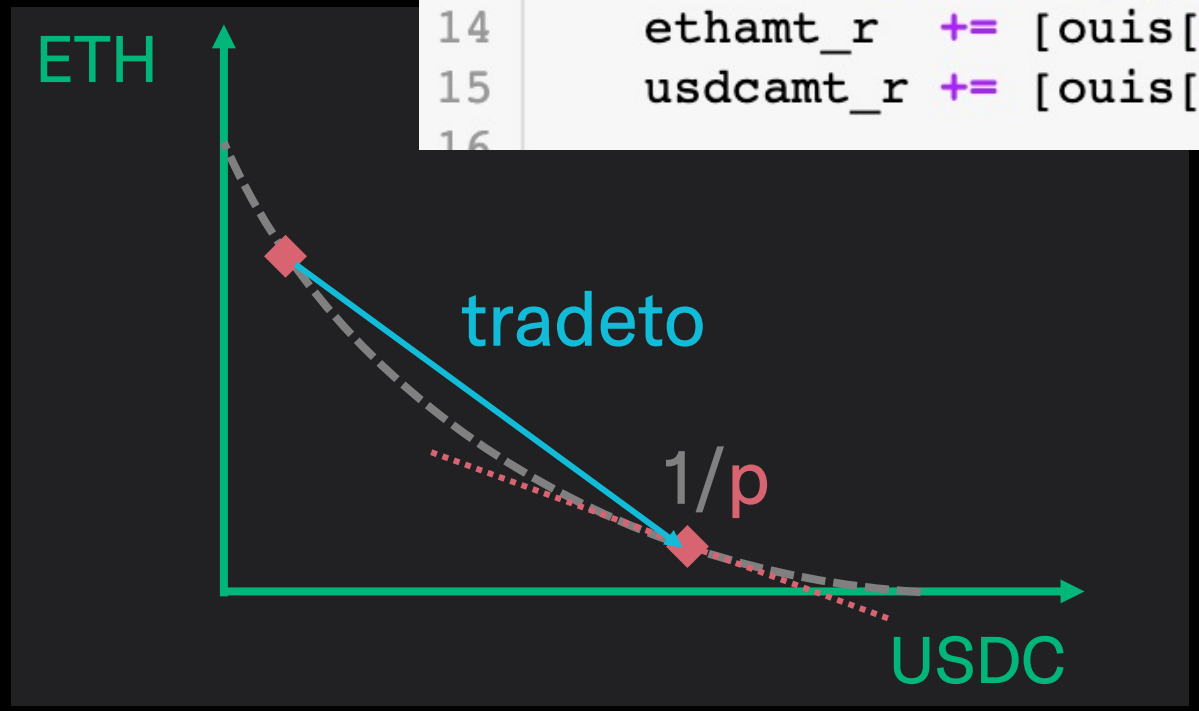
fast sim using tradeto function (Demo 7-1; NBTest 50 ,51)

Using the tradeto function

Demo 7-1

Demo 7-2

```
11 for spot in path[1:]:
12     for oui in ouis.values():
13         oui.tradeto(spot)
14     ethamt_r  += [ouis[0].y]
15     usdcamt_r += [ouis[1].y]
16
```



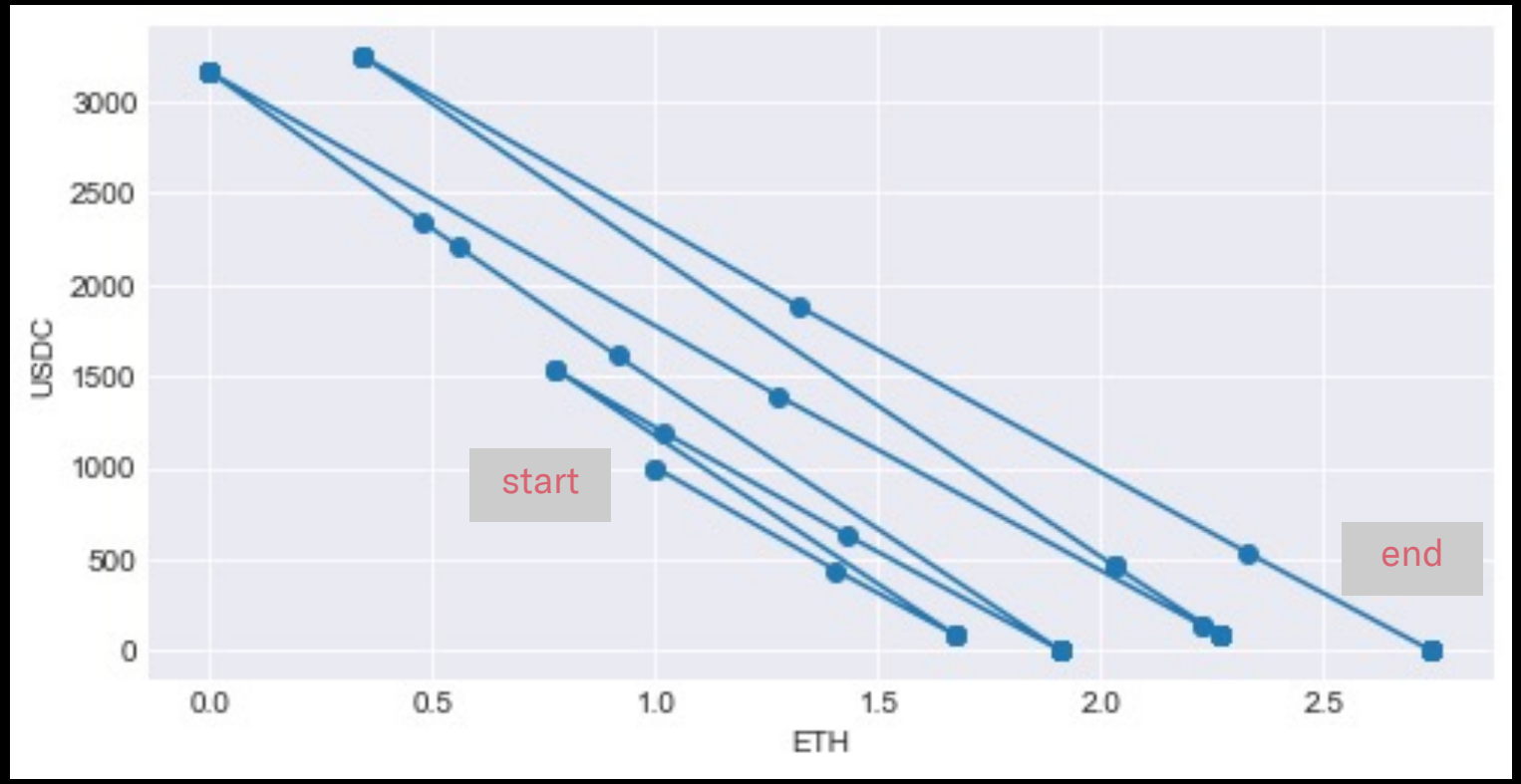
tradeto(p)

order.tradeto(p)

the tradeto method changes the state of an order, trading it to the marginal price p. the collateral received is placed with the linked order, if present

Portfolio path chart

Demo 7-1

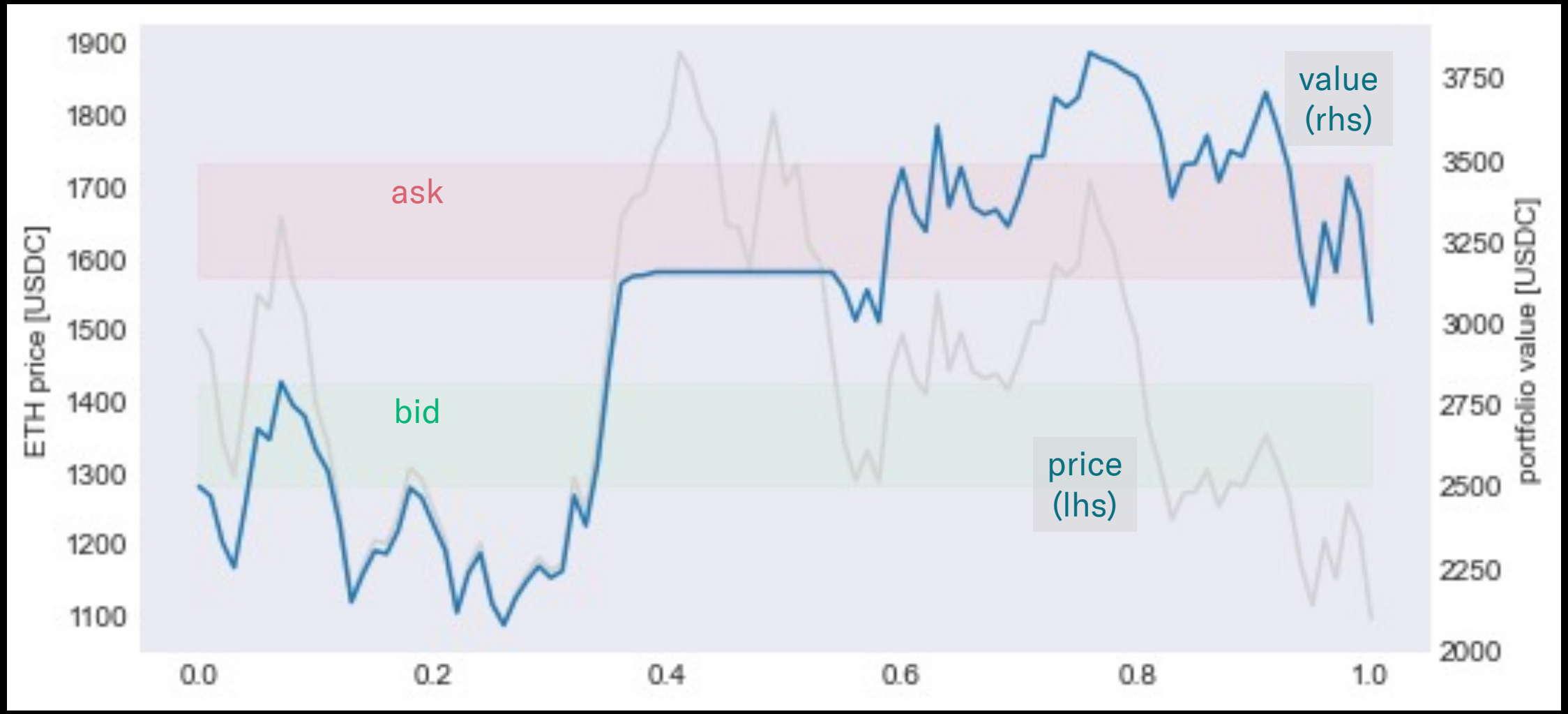


the tradeto method changes the state of an order, trading it to the marginal price p. the collateral received is placed with the linked order, if present

Portfolio value evaluation chart

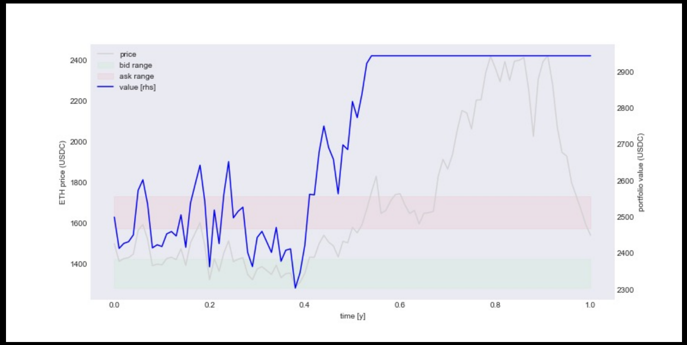
Demo 7-1

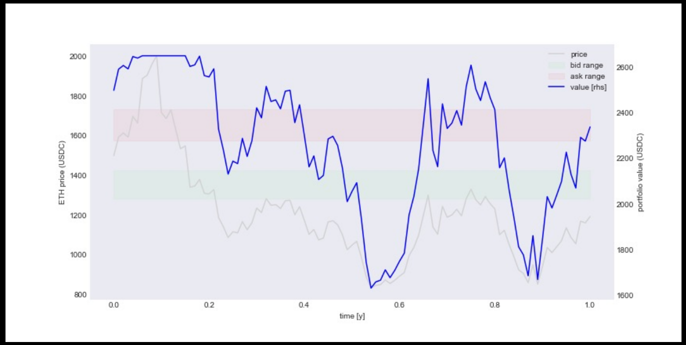
Demo 7-2

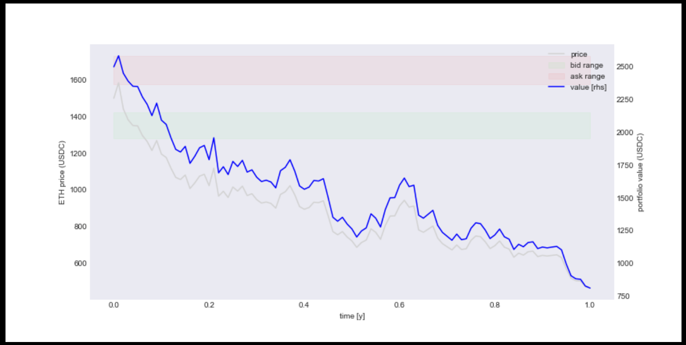


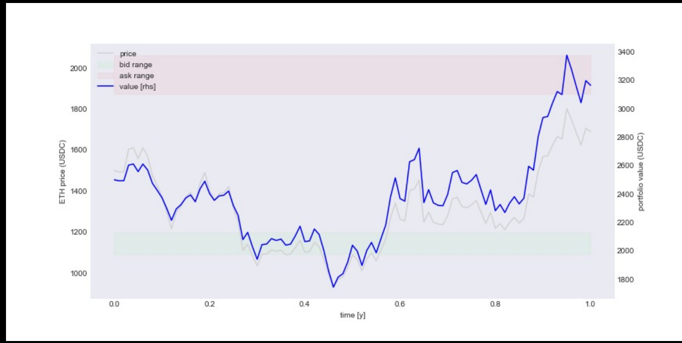
Various simulation runs

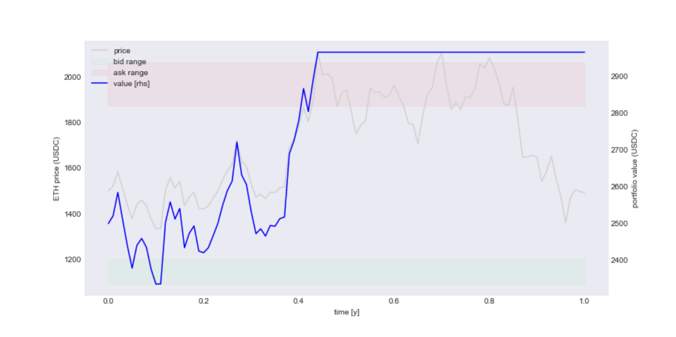
Demo 7-2

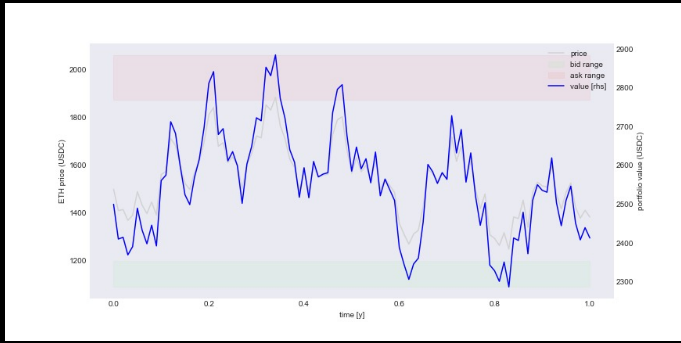


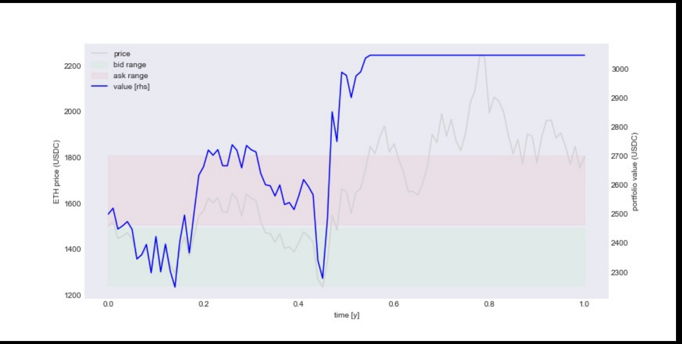


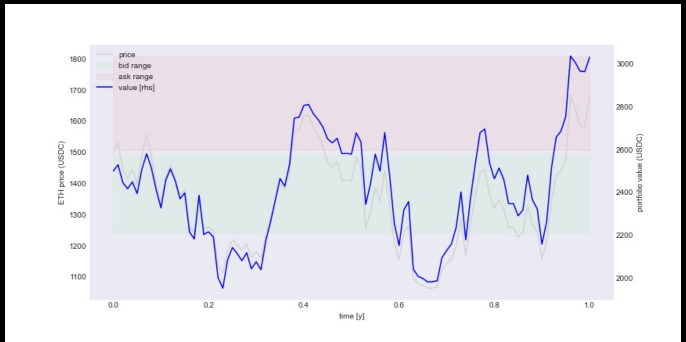


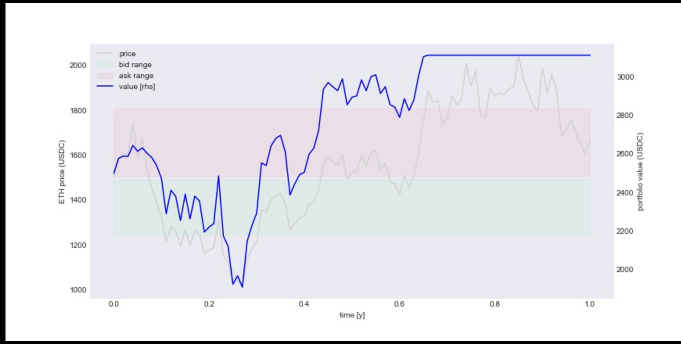












Appendix

References

- This presentation on github
- Carbon Whitepaper on carbondefi.xyz
- Carbon Simulator on github and binder
- Simulator example Demo 7-1 on github and binder
- Simulator example Demo 7-2 on github and binder

NBTest versus Demo notebooks

Demo

- Demo notebooks cover a specific use Carbon use case
- They only contain user-readable demo code, they are not part of the testing pipeline
- Demo books are roughly grouped by thematic area; eg 7-x books are about trading charts
- Demo notebooks are located in resources/demo

NBTest

- NBTest notebooks are developed in line with features of the Carbon library
- They may contain some demo code, but most importantly they contain tests
- NBTest books are sequentially numbered, by time of development of the associated feature
- NBTest notebooks are located in resources/NBTest