Why do you “need” 3 matrices?

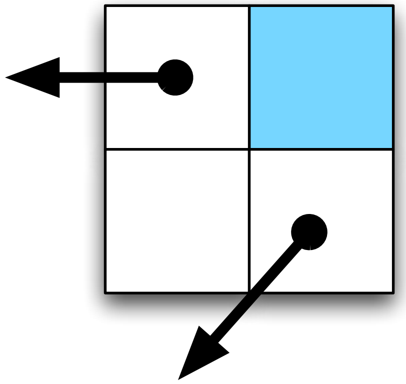
- Alternative WRONG algorithm:

```
M[i][j] = max(
    M[i-1][j-1] + cost(x[i], y[i]),
    M[i-1][j] + gap + (gap_start if Arrow[i-1][j] != ← ),
    M[j][i-1] + gap + (gap_start if Arrow[i][j-1] !=  ↓ )
)
```

WRONG Intuition: we only need to know whether we are starting a gap or extending a gap.

The arrows coming out of each subproblem tell us how the best alignment ends, so we can use them to decide if we are starting a new gap.

The best alignment up to this cell ends in a gap.



The best alignment up to this cell ends in a match.

PROBLEM: The best alignment for strings x[1..i] and y[1..j] doesn’t have to be used in the best alignment between x[1..i+1] and y[1..j+1]