We partition the matrix into 8-by-8 blocks. The blocks along the diagonal are easily handled, as in the code. Here we explain how we use 16 cache misses (the theoretical minimum) to transpose a block not on the diagonal.

In the diagrams below we show an 8-by-8 block in matrix A on the left. We show the corresponding block in Matrix B (the one we will transpose the block in A to) on the right.

Each spreadsheet cell contains the cache set number that the corresponding memory address will be assigned to. We can see that our method works only because A and B are aligned nicely in memory and have a particular size.

Matrix A

2	2	2	2	2	2	2	2
10	10	10	10	10	10	10	10
18	18	18	18	18	18	18	18
26	26	26	26	26	26	26	26
2	2	2	2	2	2	2	2
10	10	10	10	10	10	10	10
18	18	18	18	18	18	18	18
26	26	26	26	26	26	26	26

Matrix B

1	1	1	1	1	1	1	1
9	9	9	9	9	9	9	9
17	17	17	17	17	17	17	17
25	25	25	25	25	25	25	25
1	1	1	1	1	1	1	1
9	9	9	9	9	9	9	9
17	17	17	17	17	17	17	17
25	25	25	25	25	25	25	25

1. We first transpose the orange block in A to the orange block in B.

	v1	v2	v3	v4
	v5	v6	v7	v8
	18	18	18	18
	26	26	26	26

	1	1	1	1
	9	9	9	9
	17	17	17	17
	25	25	25	25

2	2	2	2	2	2	2	2
10	10	10	10	10	10	10	10
18	18	18	18	18	18	18	18
26	26	26	26	26	26	26	26

1	1	1	1	1	1	1	1
9	9	9	9	9	9	9	9
17	17	17	17	17	17	17	17
25	25	25	25	25	25	25	25

2. Now we store 8 cells in the top right of A in 8 local variables.

			v1	v2	v3	v4	
			v5	v6	v7	v8	
			18	18	18	18	
			26	26	26	26	
2	2	2	2	2	2	2	2
10	10	10	10	10	10	10	10
18	18	18	18	18	18	18	18
26	26	26	26	26	26	26	26

			1	1	1	1	
			9	9	9	9	
			17	17	17	17	
			25	25	25	25	
1	1	1	1	1	1	1	1
9	9	9	9	9	9	9	9
17	17	17	17	17	17	17	17
25	25	25	25	25	25	25	25

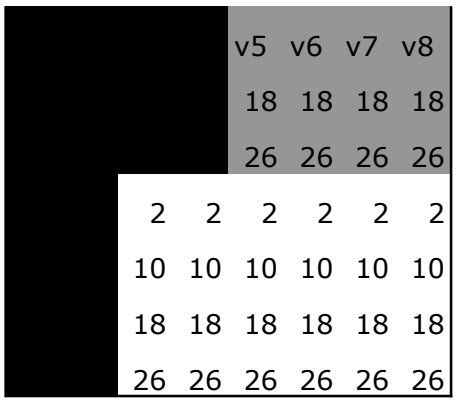
3. We move the orange block in the left to the orange block on the right.

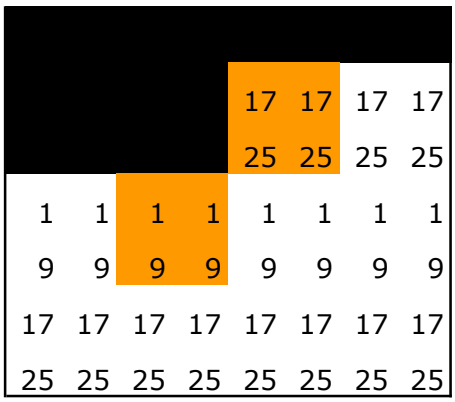
			v1	v2	v3	v4	
			v5	v6	v7	v8	
			B	B	B	B	
			B	B	B	B	
2	2	2	2	2	2	2	2
10	10	10	10	10	10	10	10
18	18	18	18	18	18	18	18
26	26	26	26	26	26	26	26

			1	1	1	1	
			9	9	9	9	
			17	17	17	17	
			25	25	25	25	
1	1	1	1	1	1	1	1
9	9	9	9	9	9	9	9
17	17	17	17	17	17	17	17
25	25	25	25	25	25	25	25

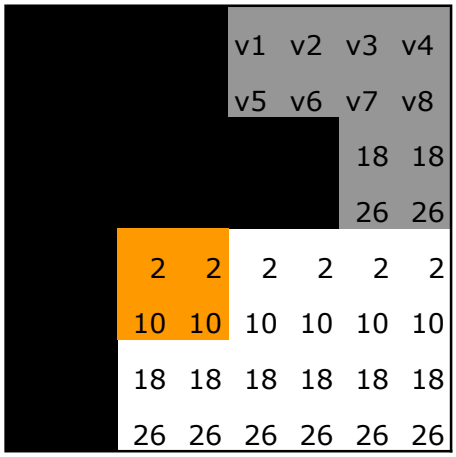
4. We transpose the orange block on the left to the orange block on the right.

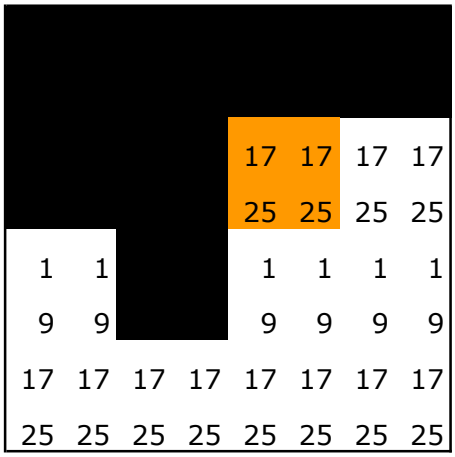
	v1	v2	v3	v4



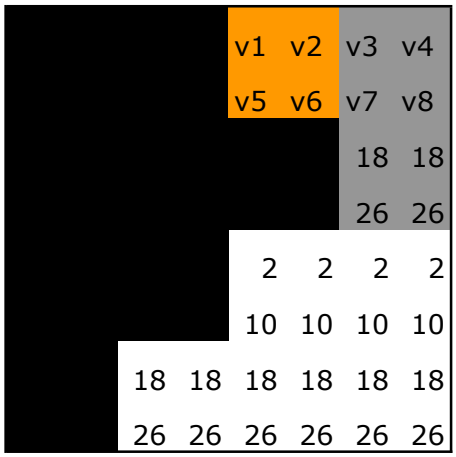


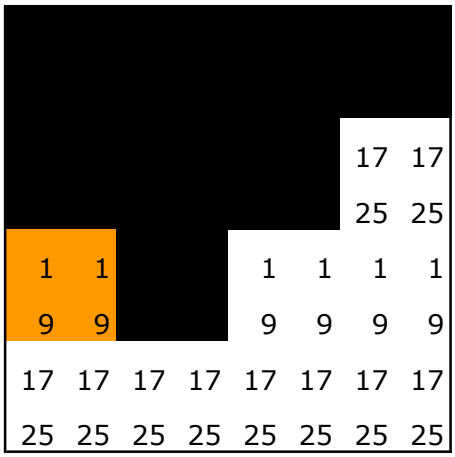
5. We move the top right orange box in B to the bottom left orange box in B (leaving A unchanged).



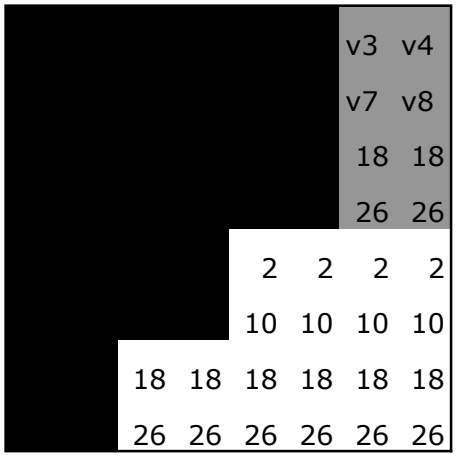


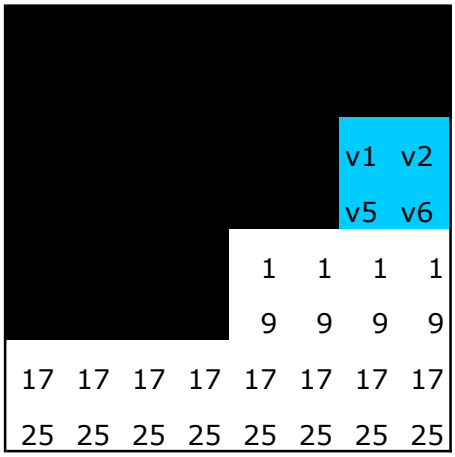
6. We transpose the orange block on the left to the orange block on the right.



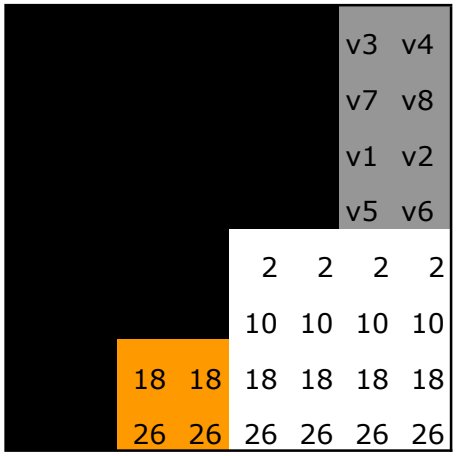


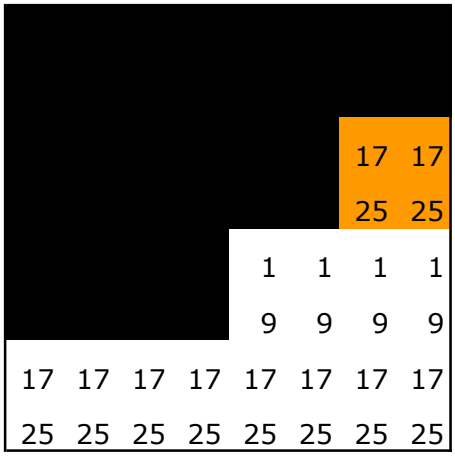
7. We transpose the orange block on the left to the orange block on the right. However, we do not access matrix A at all. Instead we use the local variables we have stored the values inside.



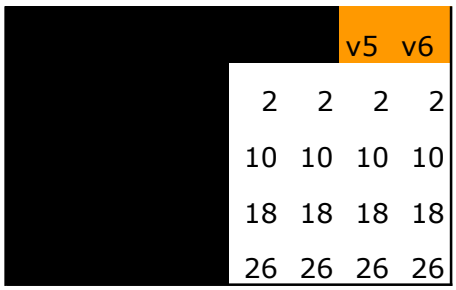


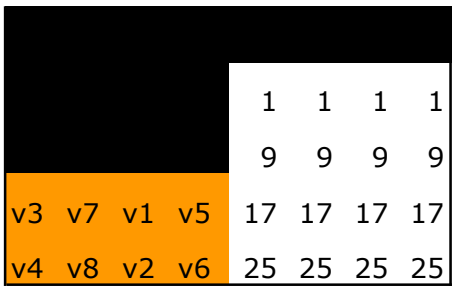
8. We store the orange block on the right in local variables.



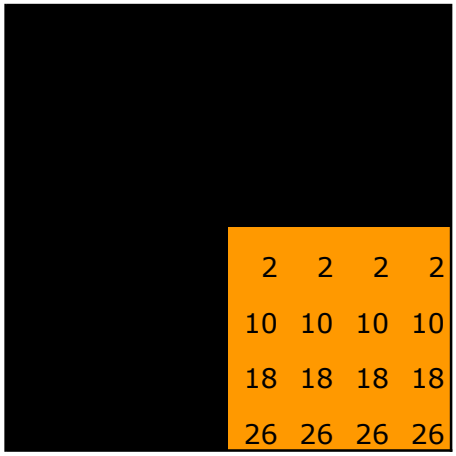


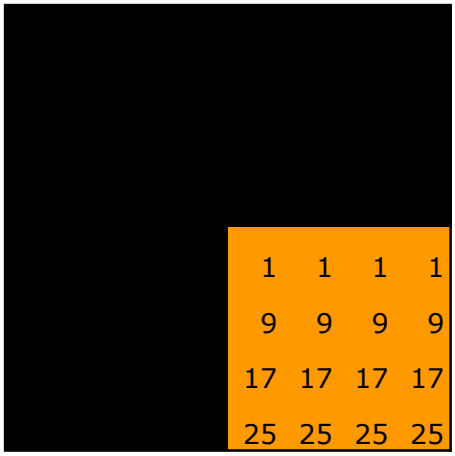
9. We transpose the orange block on the left to the orange block on the right.



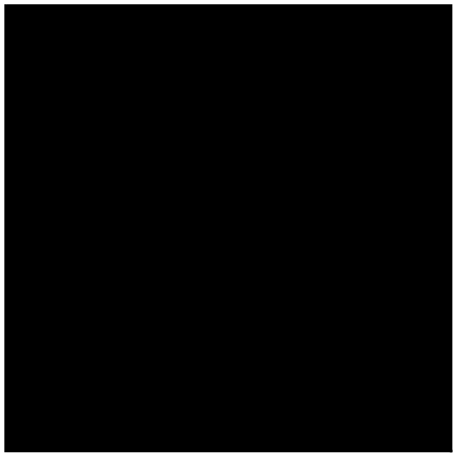


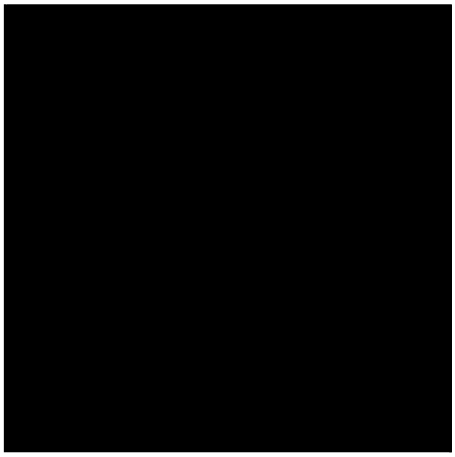
10. We transpose the orange block on the left to the orange block on the right. However, we do not access matrix A at all. Instead we use the local variables we have stored the values inside.





11. We transpose the orange block on the left to the orange block on the right.





The matrix blocks are now fully transposed. It is easy to compute that we loaded each line into cache exactly once which gives us 16 misses. Since there are 16 lines this is the theoretical minimum.