

Sparse AD

Benoît Legat

☐ Full Width Mode ☐ Present Mode

☰ **Table of Contents**

Sparse Jacobian

Sparsity detection of Jacobian

What color is your Jacobian ?

What color is your Hessian ?

Less colors but nontrivial substitution

- [What color is your Jacobian ? Graph Coloring for Computing Derivatives](#)
- [An Illustrated Guide to Automatic Sparse Differentiation](#)
- [Revisiting Sparse Matrix Coloring and Bicoloring](#)

Sparse Jacobian ⇔

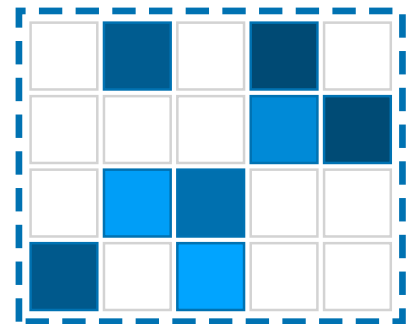
Suppose the Jacobian is **sparse**.

- The *sparsity pattern* (rows and columns of nonzeros) is **known**
- You want to determine the **value** of these nonzeros with AD

Consider the 4×5 sparse matrix on the right:

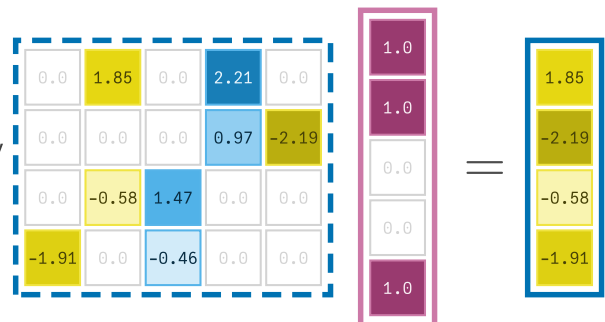
- Forward mode would require 5 JVP as there are 5 columns
- Reverse mode would require 4 VJP as there are 4 rows

Can we do better using the known sparsity pattern ?

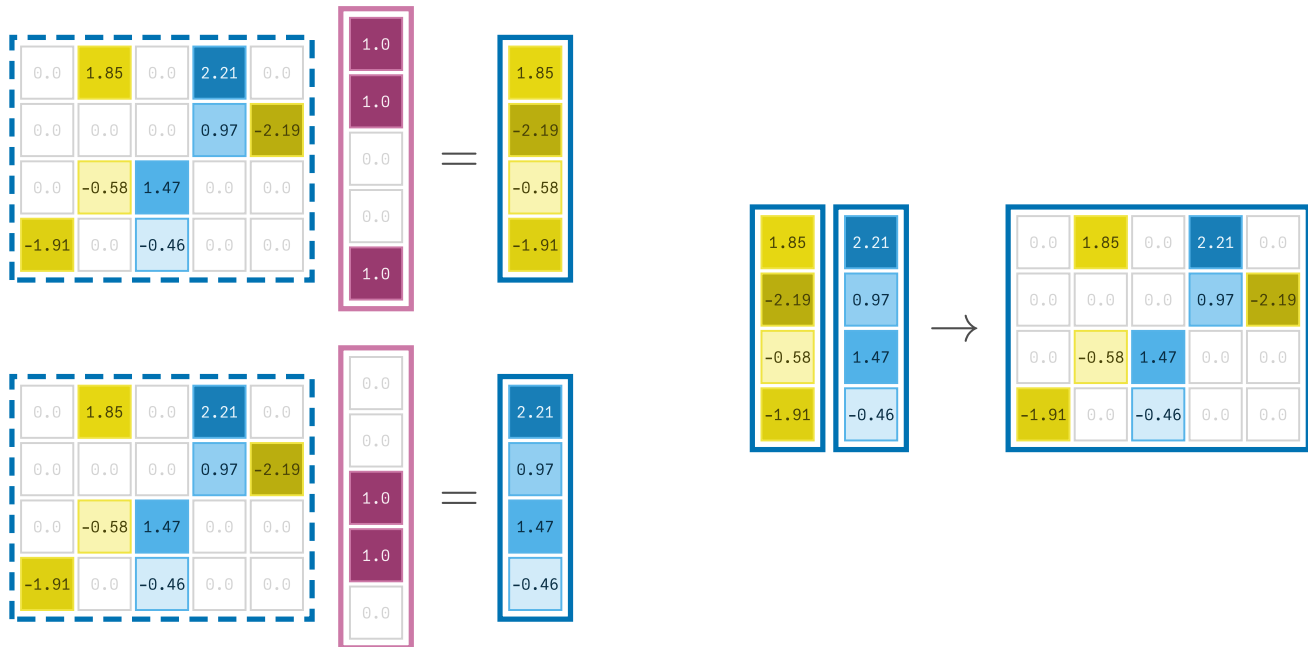


Three columns in just one JVP ⇔

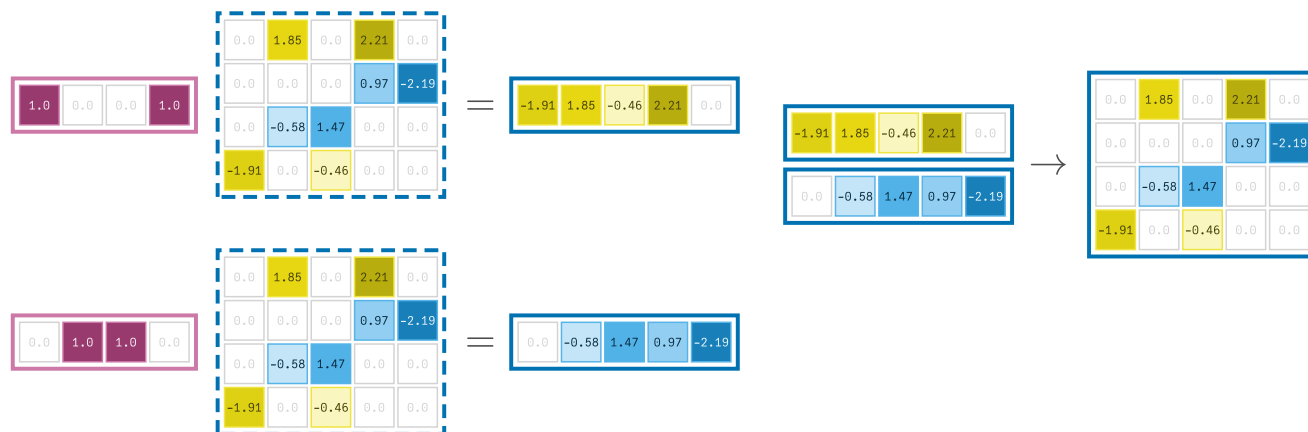
- The columns 1, 2 and 5 do not share any rows with nonzero entries.
- So their entries can be recovered unambiguously with just one JVP!



Whole Jacobian in just two JVP \Leftrightarrow



The same applies to reverse mode \Leftrightarrow



Sparsity detection of Jacobian ⇄

Simple operator overloading implementation

```
1 struct MyGradientTracer
2     indexset::Set{Int}
3 end
```

```
1 Base.:+(a::MyGradientTracer, b::MyGradientTracer) =
  MyGradientTracer(union(a.indexset, b.indexset))
```

```
1 Base.:*(a::MyGradientTracer, b::MyGradientTracer) =
  MyGradientTracer(union(a.indexset, b.indexset))
```

```
1 Base.:/(a::MyGradientTracer, b::Number) = a
```

The sign function is constant over $x > 0$ and $x < 0$ so its output carries **no** variables

```
1 Base.sign(x::MyGradientTracer) = MyGradientTracer(Set()) # return empty index set
```

Scalar example ⇄

f (generic function with 1 method)

```
1 f(x) = x[1] + x[2]*x[3] * sign(x[4])
```

xtracer =

► [MyGradientTracer(Set([1])), MyGradientTracer(Set([2])), MyGradientTracer(Set([3])), MyGradientTracer(Set([4]))]

```
1 xtracer = [
2     MyGradientTracer(Set(1)),
3     MyGradientTracer(Set(2)),
4     MyGradientTracer(Set(3)),
5     MyGradientTracer(Set(4)),
6 ]
```

ytracer = ► MyGradientTracer(Set([2, 3, 1]))

```
1 ytracer = f(xtracer)
```

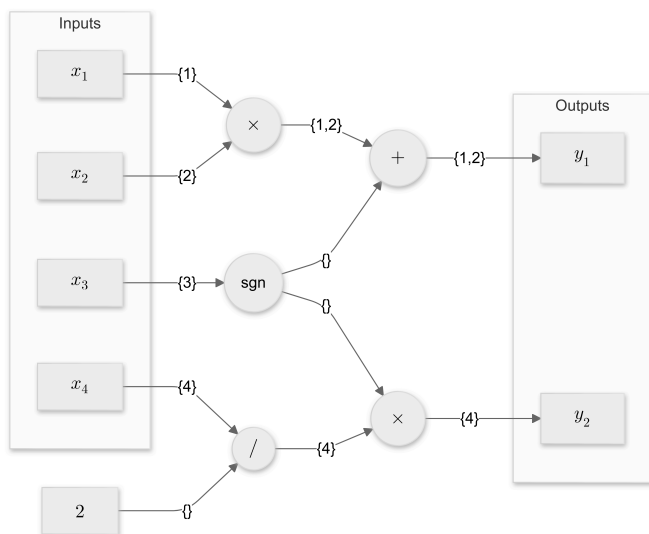
Sparsity detection with multiple outputs \Leftrightarrow

F (generic function with 1 method)

```
1 F(x) = [x[1]*x[2]+sign(x[3]), sign(x[3]) * x[4]/2]
```

```
► [MyGradientTracer(Set([2, 1])), MyGradientTracer(Set([4]))]
```

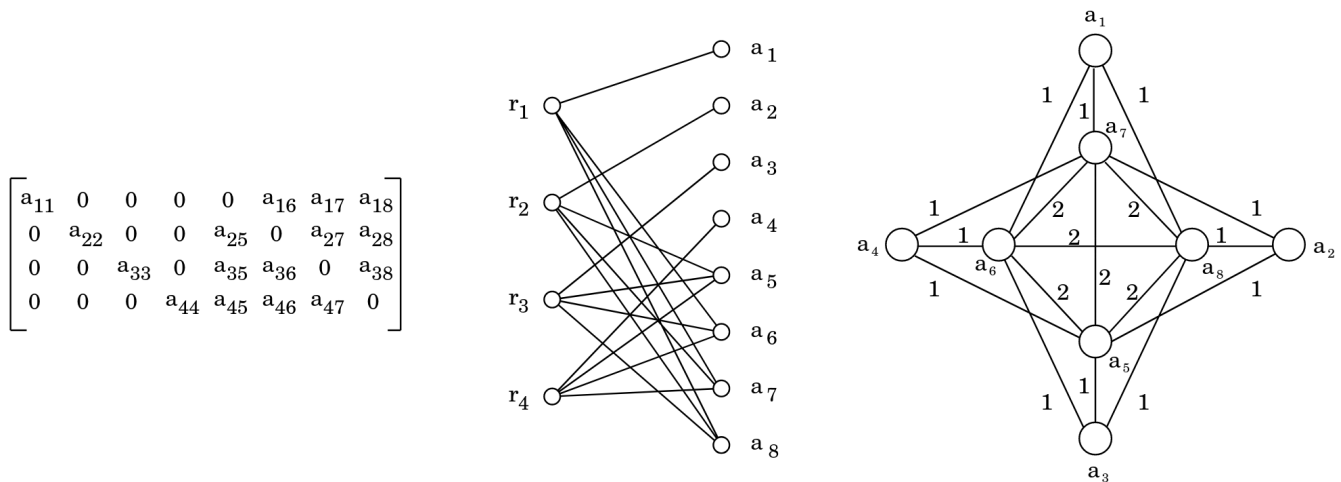
```
1 F(xtracer)
```



What color is your Jacobian ? ⇔

Given a sparse matrix A , two graphs represent its sparsity:

- Column intersection graph, used in [Software for estimating sparse Jacobian matrices, 1984](#)
- Bipartite graph, used in [Colpack](#) and [SparseMatrixColoring](#)



See [Section 3.4 of What color is your Jacobian ?](#)

Coloring problems ⇔

Given a graph $G(V, E)$,

- Nodes u, v are distance k neighbors if there exists a path from u to v of length at most k .
- A distance- k coloring : mapping $\phi : V \rightarrow \{1, \dots, p\}$ such that $\phi(u) \neq \phi(v)$ whenever u, v are distance- k neighbors.
- k -chromatic number $\xi_k(G)$: minimum p such that \exists distance- k coloring with p colors.
- Distance- k coloring problem : Find distance- k coloring with fewest colors.
- For every fixed integer $k \geq 1$, the distance- k graph coloring problem is NP-hard.

See [Section 2.1, 2.2, 3.2 of What color is your Jacobian ?](#)

Formulation as coloring problem ⇔

Theorem 3.5 A coloring of the columns is distance-2 in the bipartite graph iff columns of the same color are structurally orthogonal.

Lemma 3.7 The column intersection graph is the square of the biparted graph restricted to its column vertices.

Lemma 2.1 A coloring is distance- k in G iff it is distance-1 in G^k .

Example \Rightarrow

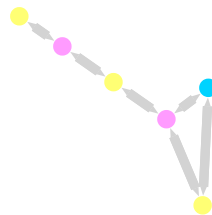
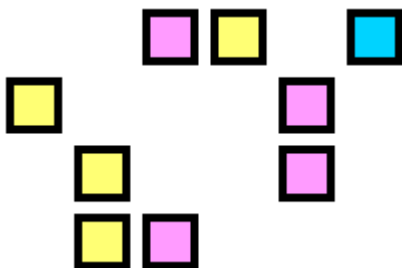
Consider the following sparsity pattern of the Jacobian matrix:

Taken from a tutorial of the [SparseMatrixColorings' doc](#)

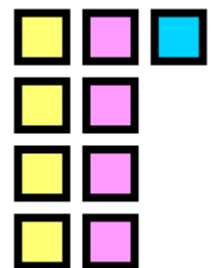
`S = 4x6 SparseMatrixCSC{Int64, Int64} with 9 stored entries:`

```
  .  .  1  1  .  1
  1  .  .  .  1  .
  .  1  .  .  1  .
  .  1  1  .  .  .
```

Color's columns are structurally orthogonal



Jacobian from 3 JVP



What color is your Hessian ? ⇔

How many HVP do we need to find the values of this sparse **symmetric** matrix ?

`ijkl` = 4x4 SparseMatrixCSC{Int64, Int64} with 10 stored entries:

```

1  2  3  .
2  4  .  5
3  .  6  .
.  5  .  7

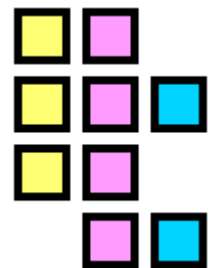
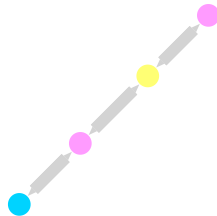
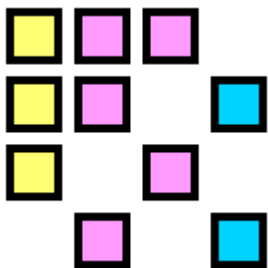
```

Section 2.4 Consider the *adjacency graph* G of a matrix A be the graph whose adjacency matrix has same sparsity pattern as A . So i and j are adjacent iff a_{ij} is nonzero.

Need 3 colors for a path of 4 vertices ⇔

$$D^T = \begin{bmatrix} \textcolor{yellow}{1} & & & \\ & \textcolor{red}{1} & \textcolor{red}{1} & \\ & & & \textcolor{blue}{1} \end{bmatrix}$$

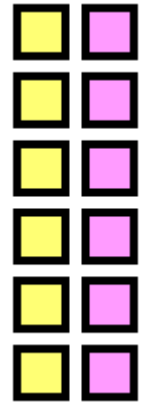
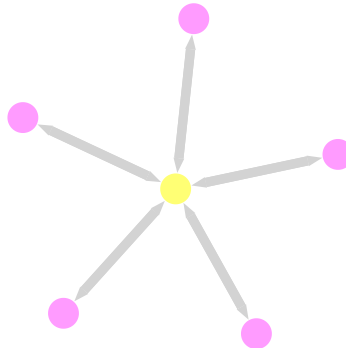
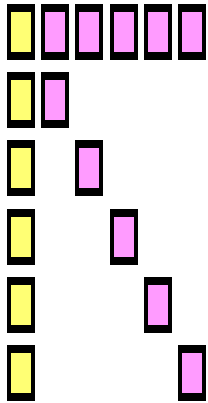
$$AD = \begin{bmatrix} \textcolor{yellow}{a_1} & \textcolor{red}{a_2 + a_3} & & \\ \textcolor{yellow}{a_2} & \textcolor{red}{a_4} & \textcolor{blue}{a_5} & \\ \textcolor{yellow}{a_3} & \textcolor{red}{a_6} & & \\ & \textcolor{red}{a_5} & \textcolor{blue}{a_7} & \end{bmatrix}$$



```
1 viz(ijkl, decomposition = :direct, structure = :symmetric, plot_size = (3cm, 3cm))
```

- Each off-diagonal entry a_{uv} corresponds to an edge in the graph.
- The edge links nodes u, v of 2 different colors $\phi(u)$ and $\phi(v)$
- If u is not adjacent to any other nodes of color $\phi(v)$ then a_{uv} is the **only term** at the row u of the HVP of color $\phi(v)$

Coloring of a star \Leftrightarrow



- For any pink node $u > 1$, the offdiagonal entry $a_{u,1}$ can be obtained from the yellow HVP.

n = 6

Star coloring \Leftrightarrow

Definition 4.5 A mapping $\phi : V \rightarrow \{1, \dots, p\}$ is a *star coloring* if

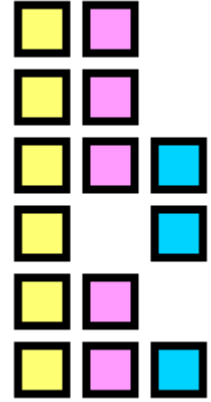
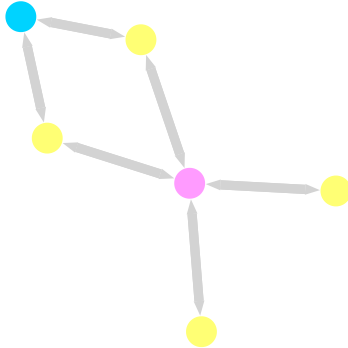
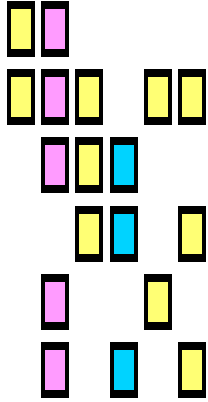
- ϕ is a distance-1 coloring
- every **path of 4 vertices** uses at least 3 colors

Theorem 4.6 Let A be a symmetric matrix with **nonzero diagonal elements**, G be its adjacency matrix. A mapping ϕ is a star coloring of the adjacency graph iff it induces a symmetrically structurally orthogonal partition of the columns of A .

Name star coloring comes from the fact that the subgraph induced by any pair of colors is a star.

► **How is the subgraph induced by a pair of colors ?**

Small Example ⇄

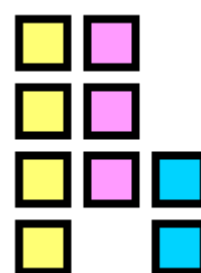
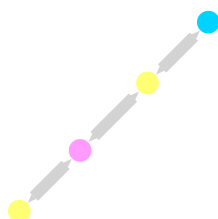
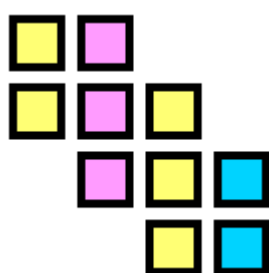


► From which HVP do we determine the off-diagonal entries ?

Less colors but nontrivial substitution ⇔

$$\begin{bmatrix} a_1 & a_2 & & & & & \\ a_2 & a_3 & a_4 & & & & \\ & a_4 & a_5 & a_6 & & & \\ & & a_6 & a_7 & & & \end{bmatrix}$$

With star coloring, 3 colors:

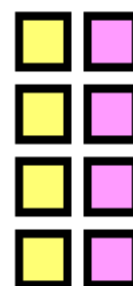
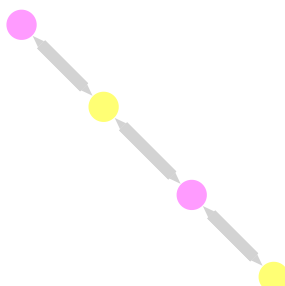
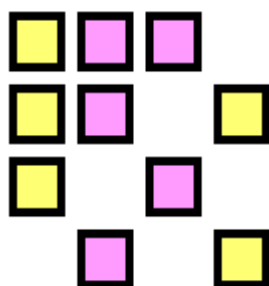


2 colors with substitutions ⇔

With 2 colors, our two forward tangents are

$$D^T = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

$$AD = \begin{bmatrix} a_1 & a_2 \\ a_2 + a_4 & a_3 \\ a_5 & a_4 + a_6 \\ a_6 & a_7 \end{bmatrix}$$



Acyclic coloring ⇔

Definition 6.3 A mapping $\phi : V \rightarrow \{1, \dots, p\}$ is an *acyclic coloring* if

1. ϕ is a distance-1 coloring
2. every **cycle** uses at least 3 colors

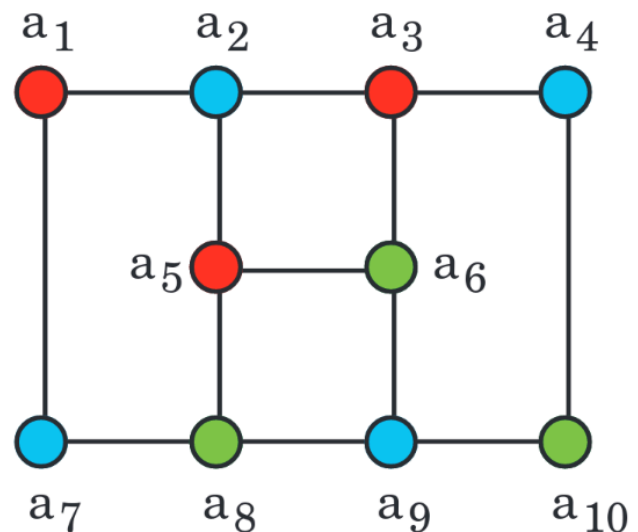
Theorem 4.6 Let \mathbf{A} be a symmetric matrix with **nonzero diagonal elements**, G be its adjacency matrix. A mapping ϕ is an acyclic coloring of the adjacency graph iff it induces a substitutable partition of the columns of \mathbf{A} .

Name acyclic coloring comes from the fact that the subgraph induced by any pair of colors is a forest.

► How is the subgraph induced by a pair of colors ?

Illustrative example ⇔

1	2	3	4	5	6	7	8	9	10
X	X								
X	X	X		X		X			
	X	X	X		X				
		X	X						
	X			X	X	X	X		
		X		X	X			X	
X				X		X	X	X	X
			X		X	X	X	X	X



For each color c , we consider the submatrix $B_c = A_{:,I}$ where $I = \{i \mid \phi(i) = c\}$ is the set of columns corresponding to color- c nodes.

► What are the nonzero entries of i th row of $B_{\phi(i)}$?

Let h_c be the sum of the columns of B_c hence the result of the corresponding HVP.

► How can the diagonal entries be determined ?

Red-Blue subgraph ⇔

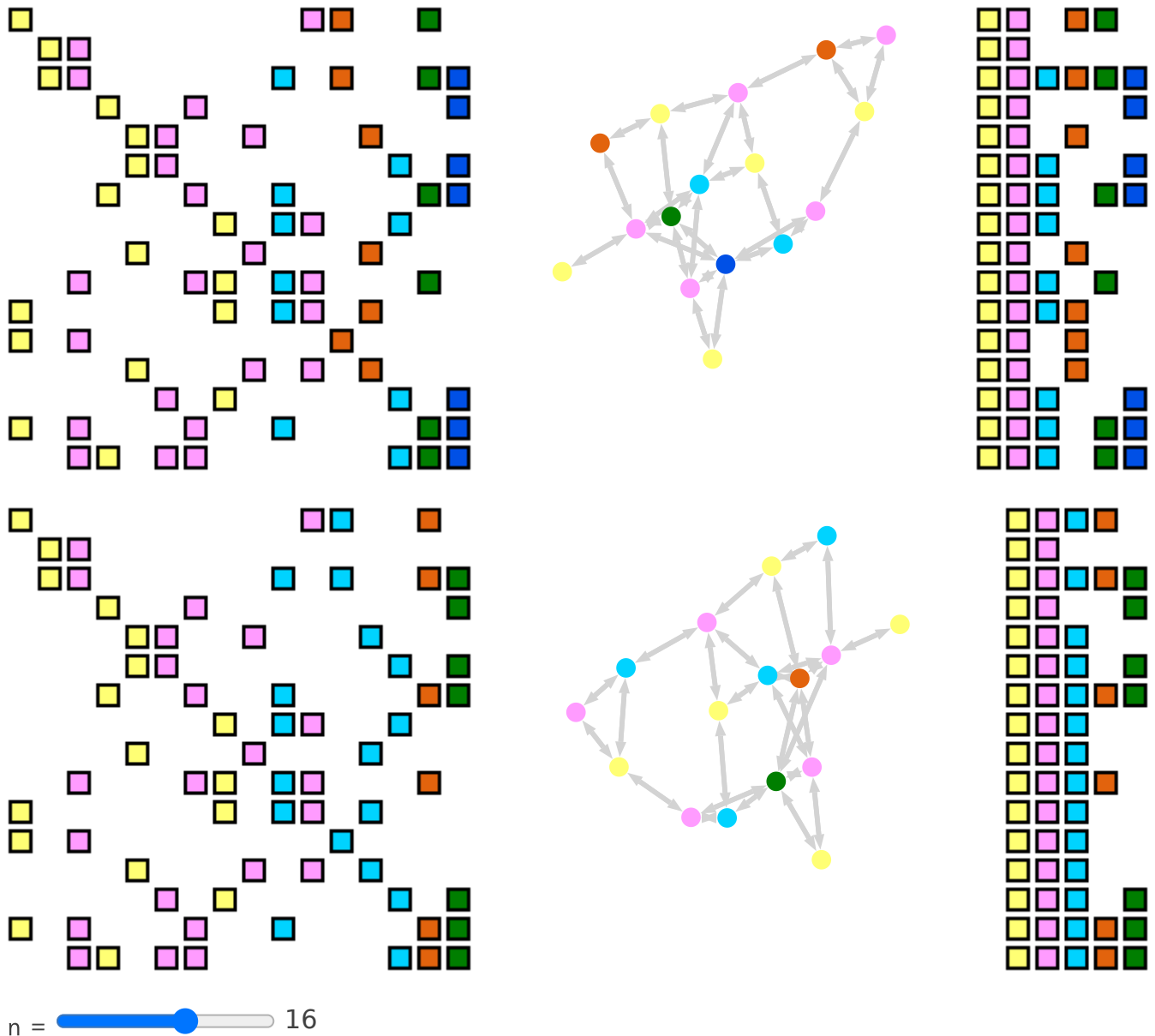
To compute an off-diagonal entry A_{ij} , consider the subgraph induced by the two colors $\phi(i)$ and $\phi(j)$. The entry will be computed, from $B_{\phi(i)}$ and $B_{\phi(j)}$, possibly after substitution.

Consider below left B_{red} and right B_{blue} . The rows corresponding to diagonal entries have been omitted as we already found how to compute them in the previous slide. The rows corresponding to green nodes are in bold as they will be computed from the subgraph induced by red/green or blue/green so we can ignore them for now.

$$\begin{pmatrix} a_{21} & a_{23} & a_{25} \\ & a_{43} & \\ & \mathbf{a_{63}} & \mathbf{a_{65}} \\ a_{71} & & \\ & & \mathbf{a_{85}} \end{pmatrix} ; \begin{pmatrix} a_{12} & & a_{17} & \\ a_{32} & a_{34} & & \\ a_{52} & & & \\ & & & \mathbf{a_{69}} \\ & & \mathbf{a_{87}} & \mathbf{a_{89}} \\ & \mathbf{a_{10,4}} & & \mathbf{a_{10,9}} \end{pmatrix}$$

► How to compute the non-bold entries ?

Larger example : star vs acyclic coloring \Leftrightarrow



Comparison of chromatic numbers \Leftrightarrow

Theorem 7.1 For every graph G ,

$$\xi_1(G) \leq \xi_{\text{acyclic}}(G) \leq \xi_{\text{star}}(G) \leq \xi_2(G) = \xi_1(G^2)$$

► What is the 1-chromatic number of planar graphs ?

Utils

scale =  pad =  border = 

```
1 using PlutoUI, PlutoUI.ExperimentalLayout, HypertextLiteral, PlutoTeachingTools
```

```
1 using SparseArrays, Images, SparseMatrixColorings
```

```
1 using Graphs, GraphPlot, LinearAlgebra, StableRNGs
```

viz (generic function with 1 method)

```
1 viz(args...; kws...) = three_columns(colored_plots(args...; kws...)...)
```

colored_plots (generic function with 1 method)

```
1 function colored_plots(A; decomposition, structure, partition = :column, kws...)
2     problem = ColoringProblem(; structure, partition)
3     algo = GreedyColoringAlgorithm(; decomposition)
4     result = coloring(A, problem, algo)
5     background_color = RGBA(0, 0, 0, 0)
6     border_color = RGB(0, 0, 0)
7     colorscheme = distinguishable_colors(
8         ncolors(result),
9         [convert(RGB, background_color), convert(RGB, border_color)];
10        dropseed=true,
11    )
12    A_img, B_img = SparseMatrixColorings.show_colors(result; colorscheme, scale,
13    pad, border, background_color, border_color)
14    if structure == :symmetric
15        S = A
16    else
17        if partition == :column
18            S = A' * A
19        else
20            S = A * A'
21        end
22    end
23    adj = SparseMatrixColorings.AdjacencyGraph(A)
24    gp = gplot(Graphs.SimpleDiGraph(S - Diagonal(diag(S)))); nodefillc =
25    colorscheme[result.color], kws...)
26    A_img, gp, B_img
27 end
```

img (generic function with 3 methods)

two_columns (generic function with 1 method)

three_columns (generic function with 1 method)

qa (generic function with 2 methods)