Steering Computational Resources

- Multidimensional Scaling
- Input: set of pairwise distances between points
- Output: a projection whose Euclidean distance best matches the input pairwise distance
- Typical objective: stress

$$Stress = \frac{\sum_{i<j}(d_{ij} - p_{ij})^2}{\sum_{i<j} p_{ij}^2}$$

MDS: Types

- Classic Multidimensional Scaling: Euclidean distances
  - Can be solved as an eigenvalue problem
- Metric Multidimensional Scaling
  - Function of Euclidean distances: often solved via stress majorization, or just gradient descent
- Nonmetric Multidimensional Scaling
  - We are not given distances, but just general dissimilarities

MDS Challenges

- Classic Multidimensional Scaling: Euclidean distances
  - Can be solved as an eigenvalue problem
- Metric Multidimensional Scaling
  - Function of Euclidean distances: often solved via stress majorization, or just gradient descent
- Nonmetric Multidimensional Scaling
  - We are not given distances, but just general dissimilarities

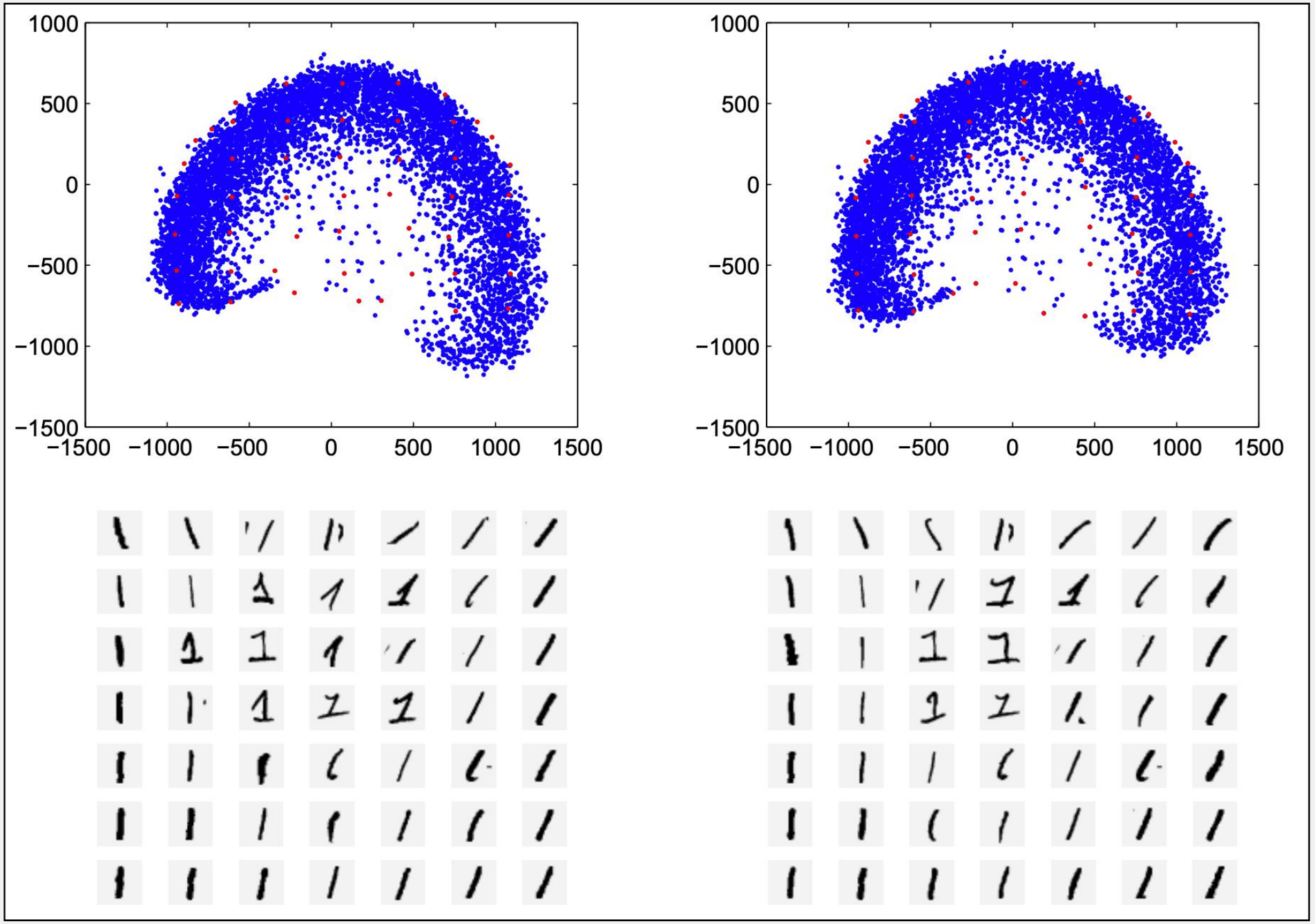
Issues with scalability

Landmark MDS

[de Silva & Tenenbaum 2004]

- Algorithm:
  - Select a small set of landmark points
  - Compute MDS on these landmarks
  - Perform scattered data interpolation to recover the rest of the points

Landmark MDS: Example



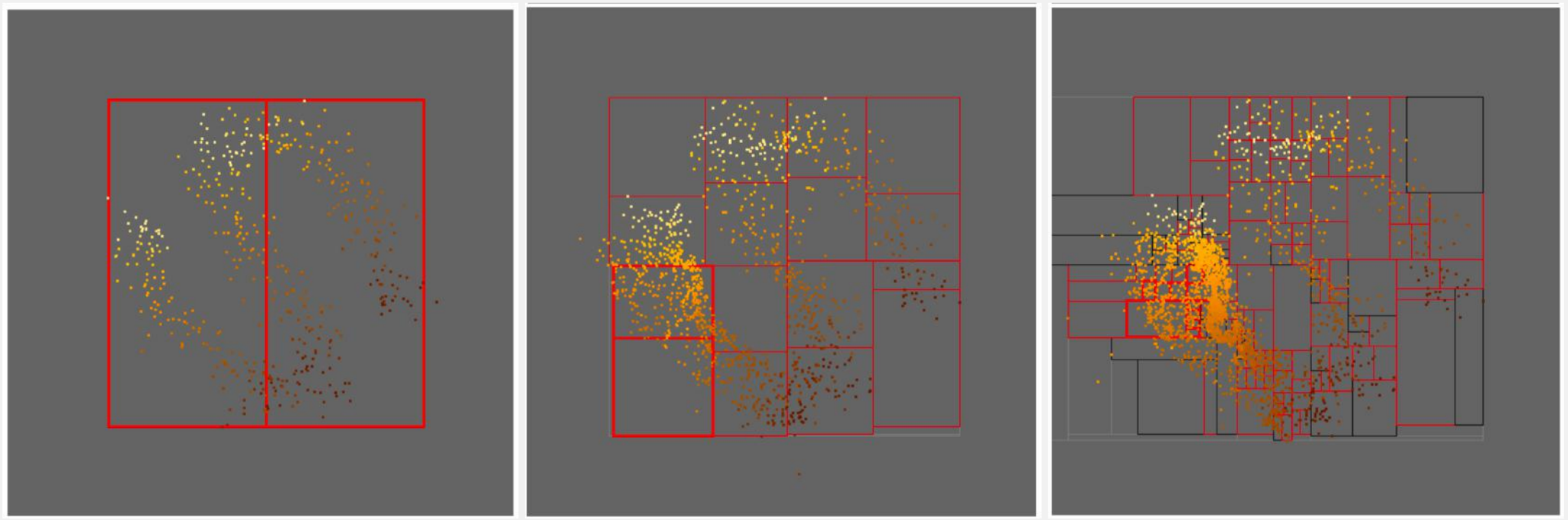
Landmark MDS: Limitations

- How do we select the landmarks?
- In [de Silva & Tenenbaum 2004], this is done automatically via farthest point sampling (or random sampling…).
- How do we have the human involved?

MDSteer

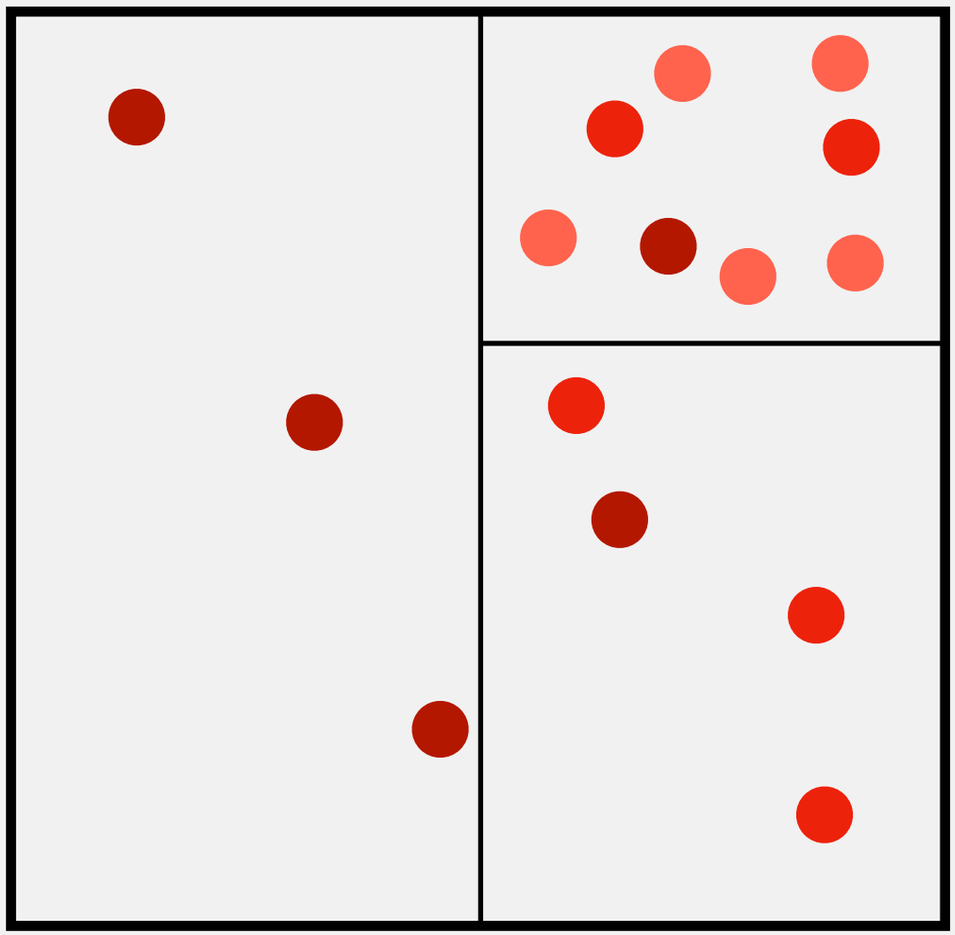
[Williams & Munzner 2004]

- Progressively perform MDS, having the user *steer the points with which to perform optimization*.



MDSteer: Bins

- Progressively subdivide the 2D plane on which the points are to be placed



- User controls what bins to subdivide
- If they do nothing: full MDS
- But, as computation progresses, they can observe, and adjust which bins to subdivide

MDSteer: Example