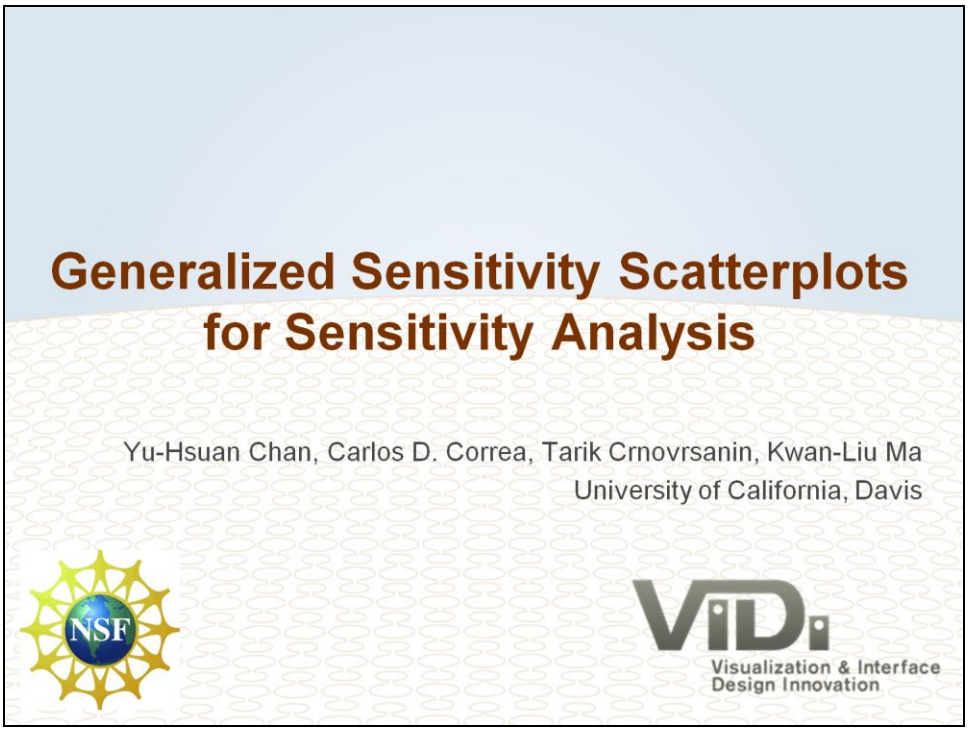
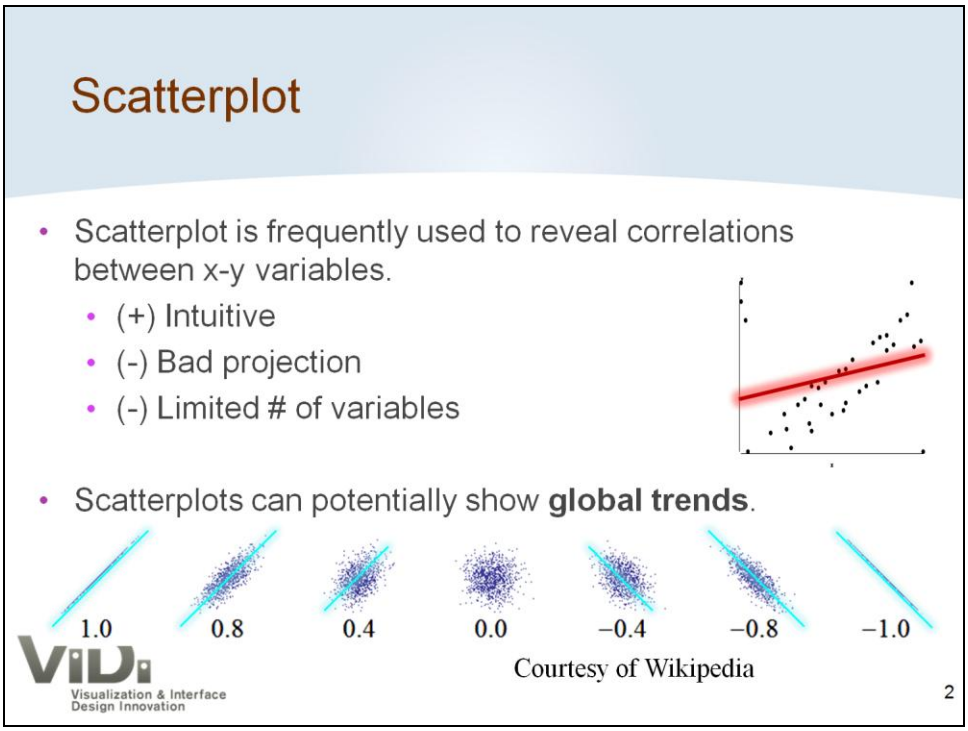


This research aims to present a new way of visualizing multi-dimensional data using generalized scatterplots by sensitivity coefficients to highlight local variation of one variable with respect to another.

These sensitivities in scatterplots can be understood as velocities, and the resulting visualization resembles a flow field.

We also present a number of operations and visual widget that help users navigate, select, cluster and analysis points in an efficient manner.



to reveal correlation between two variables.

…. approximates a positive diagonal, there is a strong positive correlation

…… a negative diagonal there are negative correlation between the two.

+ Intuitive: to study the relationship between 2 variables

-Bad projection: clutter or overlaps (correlation is hidden)

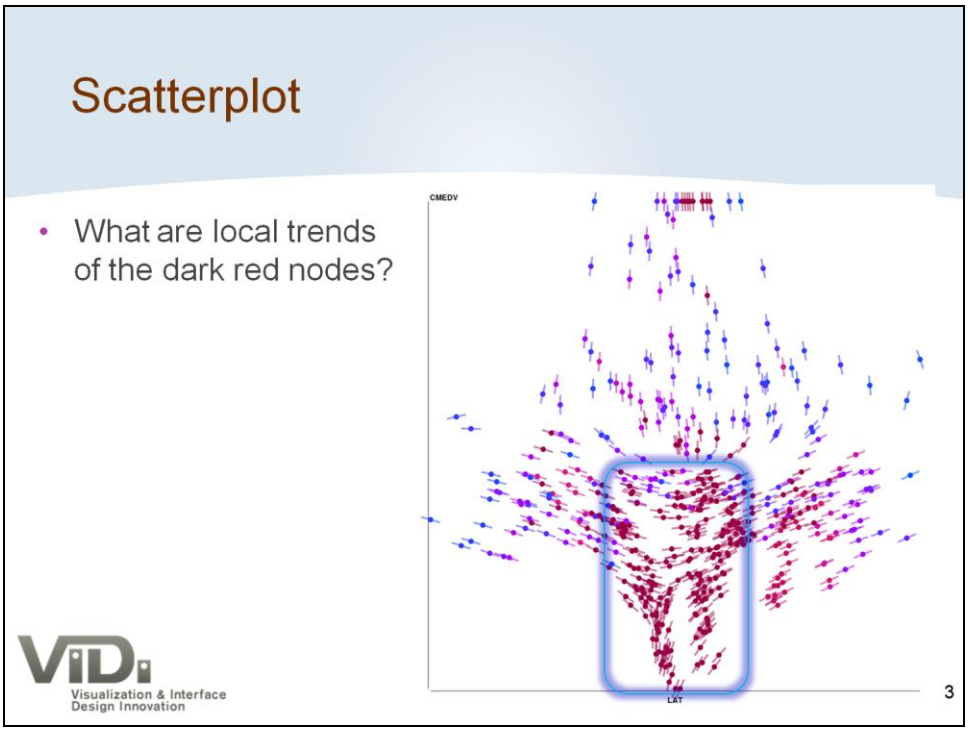
- Limited # of variables: x, y, node color, node size…….

We can use scatterplots to show trends.

<CLICK>The typically approach is to show a **single line** that shows the global trend.

<CLICK>The **problem** with this is that it is **sensitive to outliers**.: red line inclined to out liers at top left.

In this work, we propose to add the local trend to the scatterplot, and **we borrow ideas from flow visualization.**



So how can we show **local trends** in a scatterplot?
Let's look at the scatterplot on the right.
What's the local trends for those **dark red nodes at the bottom of the plot**??
<wait>
Because of the insufficient sampling, they look like in a positive correlation in this projection
We may first **subjectively** see that they all seems to be an **increasing trend.**
This this hypothesis true? We don't know.

We can not simply **put a long global trend line** on each point, because the plot becomes too **crowded** and lines would **cross** each other.
Therefore we use a **short trend line** for each node.
And the real local trends can be revealed **explicitly**.

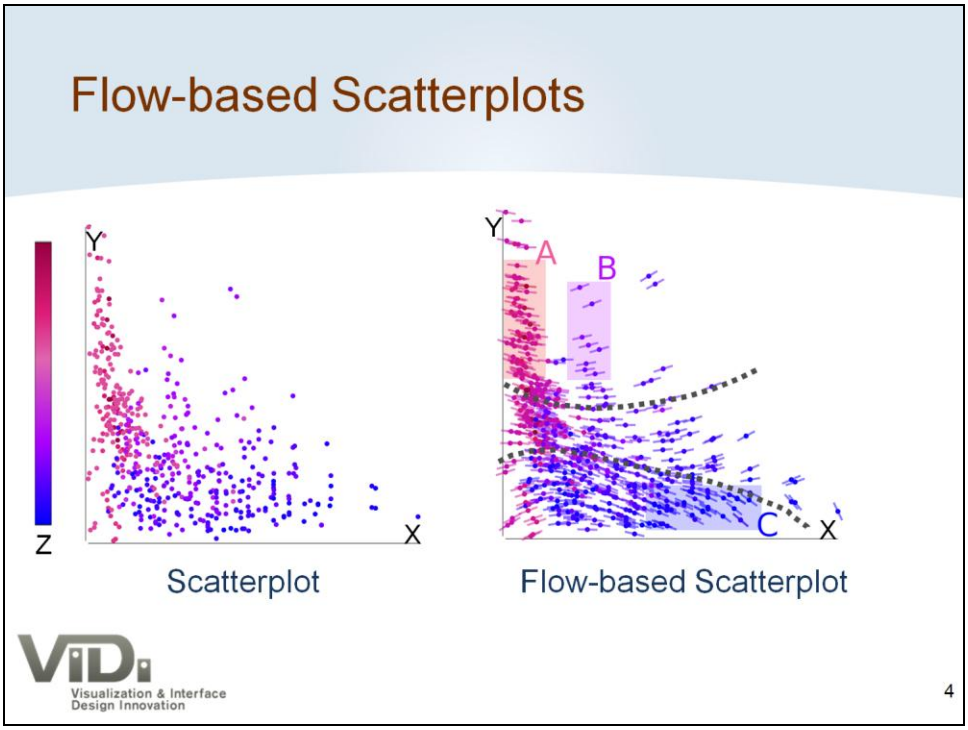
For scatterplots **without local trend information**, users can **guess** the trend.
But **outliers** or **overlap** may **obscure** the trend and often times we can be wrong.
Also we might be mislead by visual representation, like the in this case we are misled by the **node color**.

In our work, we want to see the local trends.
an approach that is **general** enough to **explore trends from local to global**.

In our approaches, we augment scatterplots with several visual augmentations, such as sensitivity lines, and streamlines
Our approach **generalize the trends**, so you can go **from local to global**.

We also make use of sensitivity to evaluate the **smoothness** of streamlines in a plot,
and therefore we have a **quantitative evaluation for projections** in multi-variate data.

We proposed several **sensitivity views** and **supported operations**
to help user efficiently navigate high dimensional space and better understand the interplay between variables.

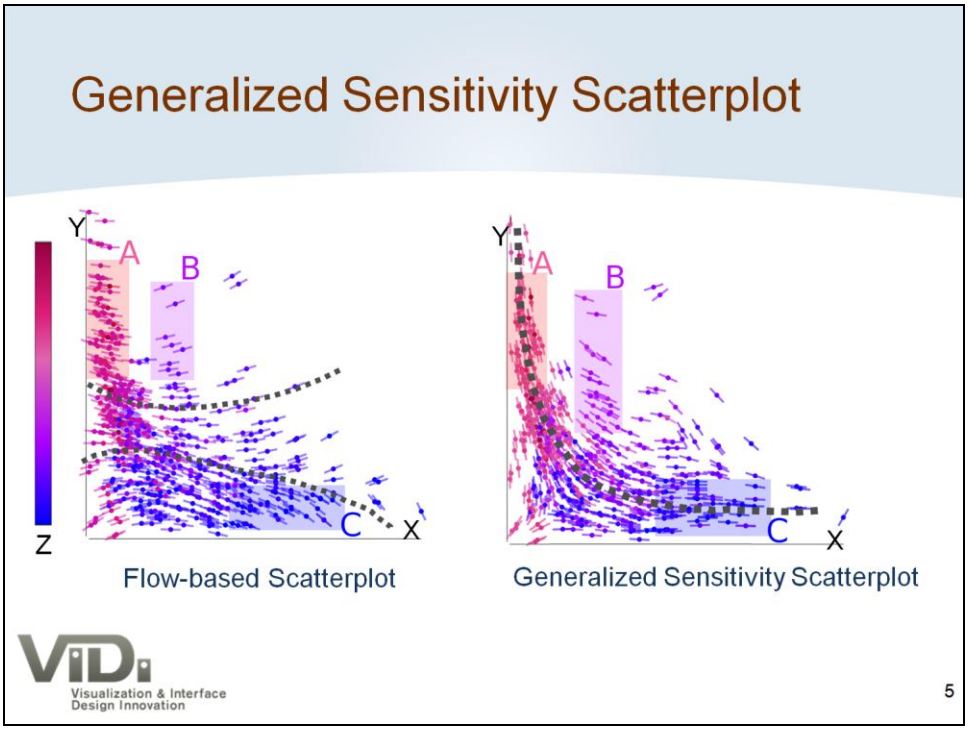


And therefore, we first proposed “flow-based scatterplots” to show local trends on a 2D scatterplot.

With traditional sccatterplots we can infer the local trend, where as with Flow-based scatterplots those sensitivity lines make the local trend explicit and useful.
Flow-based scatterplot captures and explicitly reveals correlation between Y and X locally.
Sensitivity is a high dimensional data because it has all possible combinations of 2 variables.
By short trend lines we **keep sensitivity in the plane**.

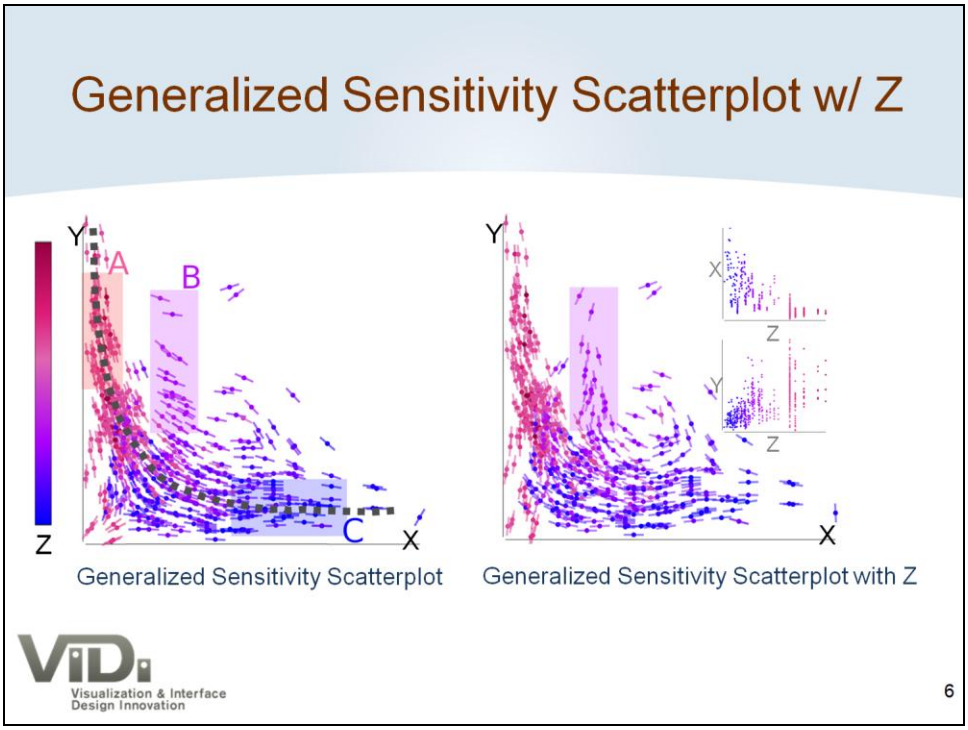
The simplest approach is to draw short lines by sensitivity computed by the 2 projection variables.
In this way, we are showing local trends of pair-wise correlation explicitly.
It is **straight forward**. You can easily see their correlations.
(A,B, C) E.g. there are 3 different **local** trends in the plot, marked as A, B and C.
(two gray dotted curves)Furthermore, you can see how data flow in the plot if you **visually connect** these short trend lines.

[[**summary**]] **sensitivity lines =** another visual attribute. We can keep sensivitiy in a plane.
Short lines in scaterplots show **relationships between two particular variables**.



Then we exploit a different way to approximate sensitivities.
(Tarik you can read this http://mathworld.wolfram.com/LeastSquaresFitting.html for details.)
In Flow-based scattter plot, the slopes of the sensitivity lines are estimated by **Vertical-offset Least Squares Fitting**,
in which a short sensitivity line on a focus node is approximate by least squares fitting with a set of neighbor nodes around it.
Lease squares fitting minimizes the sum of the squares of the offsets of the points to the line.

We tried **Perpendicular offsets in Least Squares Fitting**, as shown in the generalized sensitivity scatterplots in the right.
In comparison, **perpendicular offsets** approximate the sensitivity derivatives **better** than the vertical offsets,
especially for those nodes that Y is **very sensitive to tiny change in X** (very large slope) [A region in the LEFT and in the RIGHT].

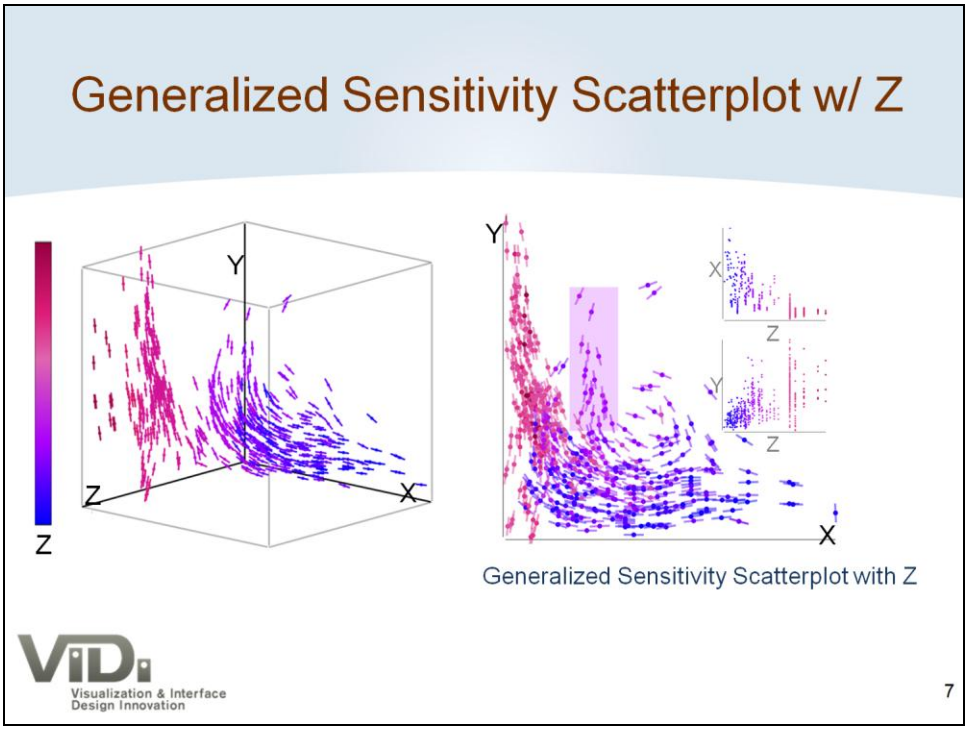


Then we also exploit the possibility to introduce more dimension onto a 2D scatterplot by sensitiviity lines.

We generalized flow-based scatterplots for more informative visualization of sensitivity.
Compare to the previous one, **Generalized Sensitivity scatterplot** (GSS) **generalizes the notion of sensitivities** with a subtle but fundamental modification:
Flow-based ones are limited to the variables involved in a 2D projection, i.e. *differentiation occurs after projection*.
In a more general sense, sensitivities could involve **the full parameter space** or **selected subspace**.
We can make use of these generalized sensitivities in a 2D plot by inverting the order of the data transformation, i.e. *projection occurs after differentiation*.

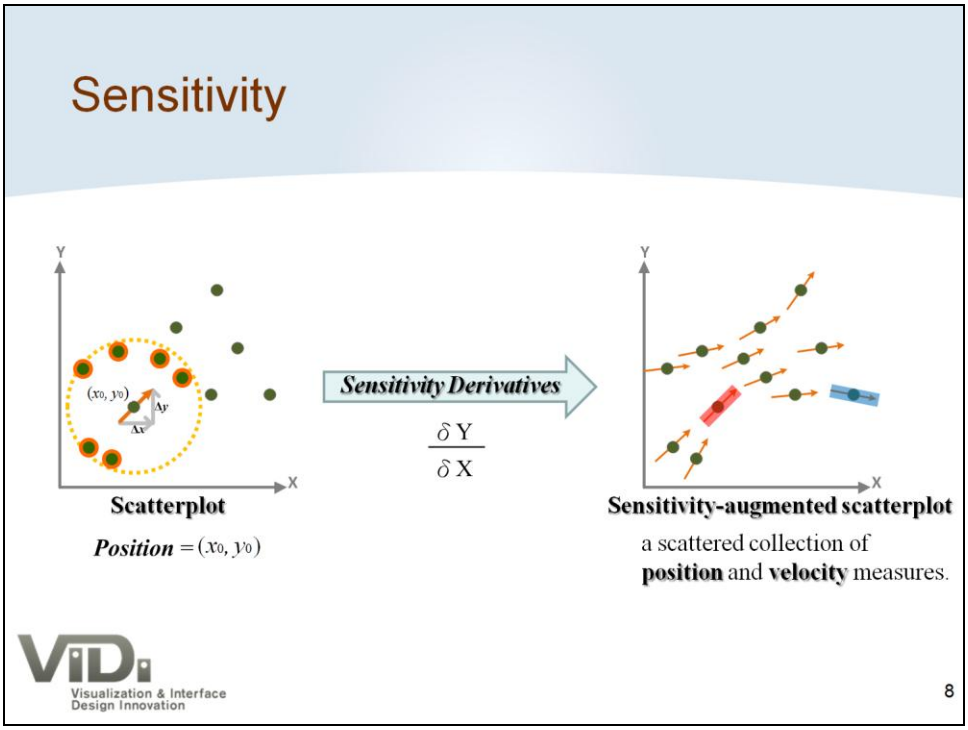
Here we show an example of GSS that sensitivties involve one more dimension **Z**.
Before this generalization, users can only use other visual metaphor like color to embed one more dimension,
like in the scatterplot in the left, nodes and deri lines are colored by the Z dimension.

We also consider Z dimension in least square fitting that uses a neighborhood to fit a short sensitivity lines on each node to represent the local trend of Y respect to X.
As shown in GSS with Z picture in the right, nodes that are far away from each other in the Z dimension can be differentiated as well.
Here we can see from the two small scatterplots of Z with X and Y on the top right, there are three main section in Z dimension, Pink, Purple and Navy blue.
The sensitivity lines consider (X,Y,Z) dimensions can reveal different local trends among these three regions from Z dimension.
Look at B region, now these purple nodes are differentiated from the pink and navy blue local trends.
This would be more clear if we show these sensitivity trend lines in a 3D cube of X,Y, Z like the following slide.
<CLICK to the next slide>



In the cube in the left, we can see the local trends are differentiated in Z dimension too.
In 3D we can clearly see the red nodes have high sensitivity of Y over X (lines with large slope).
Purple nodes seems to blend a curved hyperplane;
Navy blue nodes are those Y is insensitive to the change in X.

Here we show that **introducing 3rd** variable into a **2D scatterplot** by sensitivity lines reveals interesting correlation that vary locally from region to region.
In this way we can embed info about higher subspace onto a 2D plan, and thus we might make queries or formulate hypothesis about 3rd vari.



I will explain how to compute the sensitivity that we used in this work.

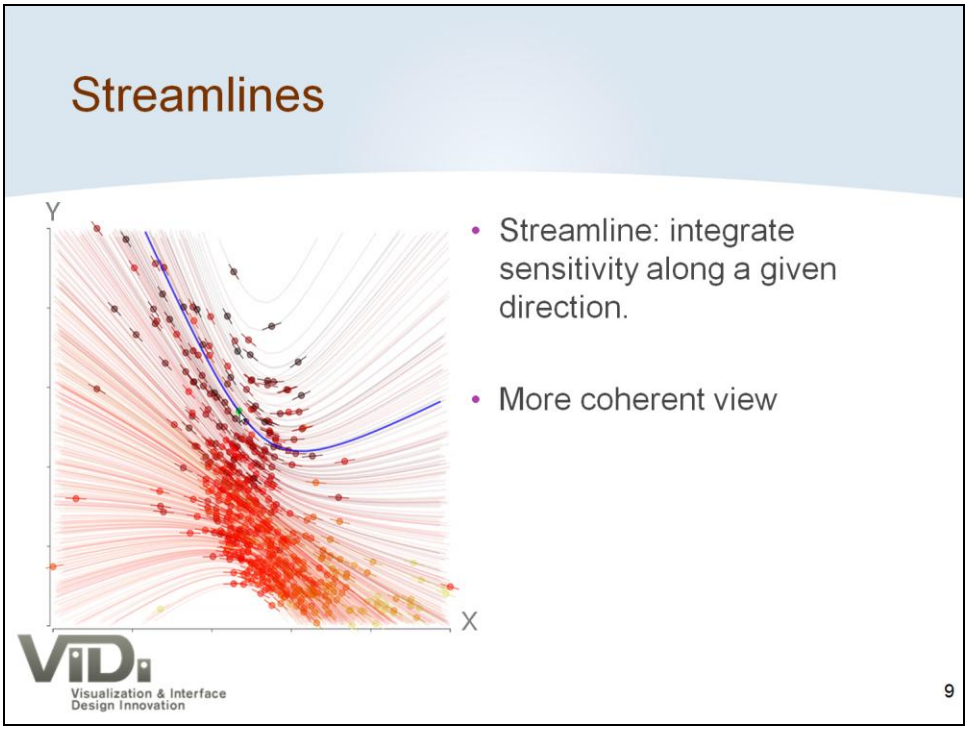
Sensitivity in general refers to how much the change in the output variable
if there is a change in the input variable.
The more the output variable change ……. , the more sensitive the output is to the input variable.
In X-Y scatterplot, sensitivity means how much change in Y result from the change in X.

We compute **sensitivity of each data point** by **approximating** the **derivative** of Y over X **at a given neighborhood**.
Derivatives are approximated by **linear least squares fitting** by **a set of neighboring nodes** around the focus point.
And we plot sensitivity lines whose **slope equals the derivatives**.

<CLICK> NOTE: a set of **neighboring nodes**
summing up the change in Y respect to the change in X.
each sensitivity line **summarizes the change locally**.

<CLICK>
/ (red) it means Y increases when X increases, and therefore in this local region Y is positive related to X.
\ (blue) it means Y decreases when X increases, and therefore in this local region Y is negative related to X.
Sensitivity lines shows:
the **magnitude** of the sensitivity by the slope;
the **positive** or **negative** correlation by the direction of the lines.

These sensitivity lines reveal the local trends explicitly.
When we look at these sensitivity lines in a plot, we could obtain a sense of the ***flow*** *of the data trend*,

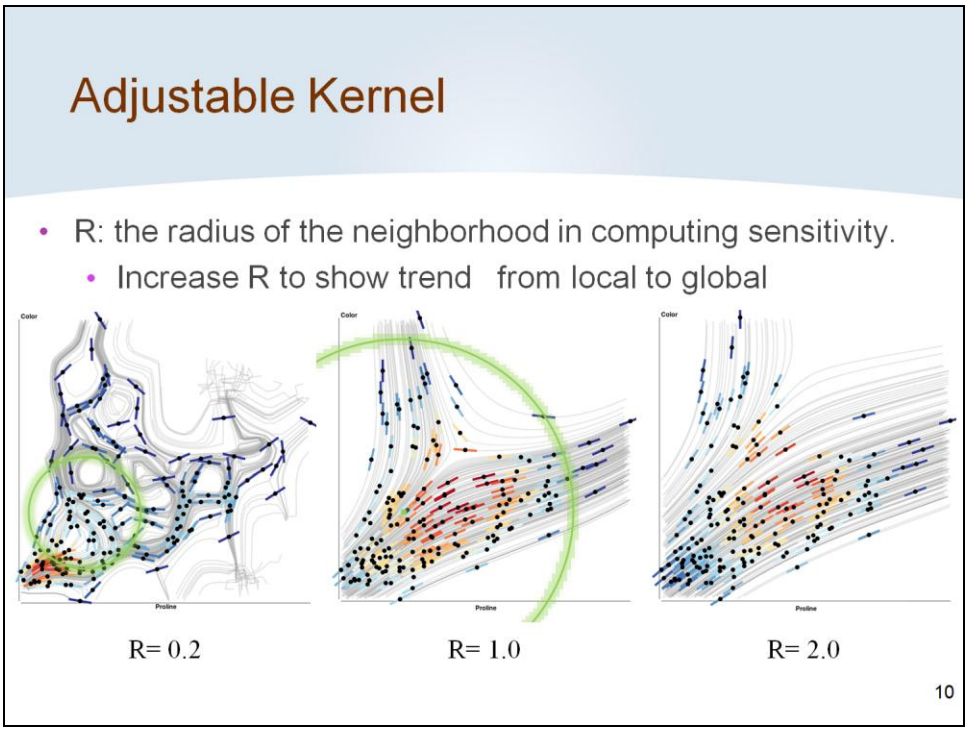


Then we borrow **idea from flow visualization**
to **integrate** local sensitivity trend into **streamlines**.

The reason WHY we can use flow to visualize correlation?
The **location** of a data point in X-Y represent **position**,
then the sensitivity **derivatives** can be treated as the **velocity** at that position.
So we integrate velocity to get streamlines.

NOTE we don't compute streamlines for sensitivity derivatives involving the third variable X, as I mentioned in the previous slide.
We only compute streamline for sensitivity of Y over X in an X-Y scatterplot.
Because streamlines that integrate **sensitivity of variable other than projection variable** my be misleading and meaningless on the X-Y scatterplot.

Streamlines enable us to look at scatterplot **in a sense of flow easily**.
And we can do some non-linear operations **operations** by these streamlines, which I will talk about later.



Remember that sensitivity is computed by **least squares fitting** by **a set of neighboring nodes.**

So we have an adjustable kernel R to decide the size of the neighborhood.

A small R use a small neighborhood, and therefore sensitivity lines reveal trends of a small local region. Sensitivity lines and flow lines have plenty of twist.

In the opposite if we use a large R, the sensitivity derivatives are computed considering a large region and therefore show global trend.

We can see the sensitivity derivatives of different neighborhood size reveal different trend.

The approximate region of a neighborhood is marked by the green circles.

When R=2.0 the whole projection space is in the neighborhood, and it reveal global trends.

User can change the kernel size according to what kind of trend they are interested in.

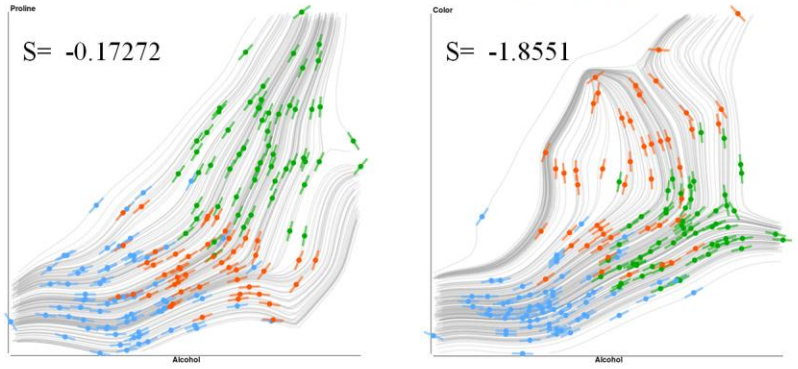
Note that the color of sensitivity lines indicate the local density of nodes in the neighborhood: red for very dense area and navy blue for sparse area.

Smoothness of a Projection

- Regions with large variance in sensitivity are not smooth.

C_i : complexity of a node i

S : smoothness of the projection $= (-1)(\sum_{i=0}^{N} C_i)$



11

We also make use of sensitivity info to **quantify the complexity of a projection.**

a quantitative evaluation.

We compute the **complexity of a node Ci** in terns of the **total variation of sensitivity derivatives in the neighborhood**,

And then accumulate for all nodes to evaluate the smoothness of a projection.

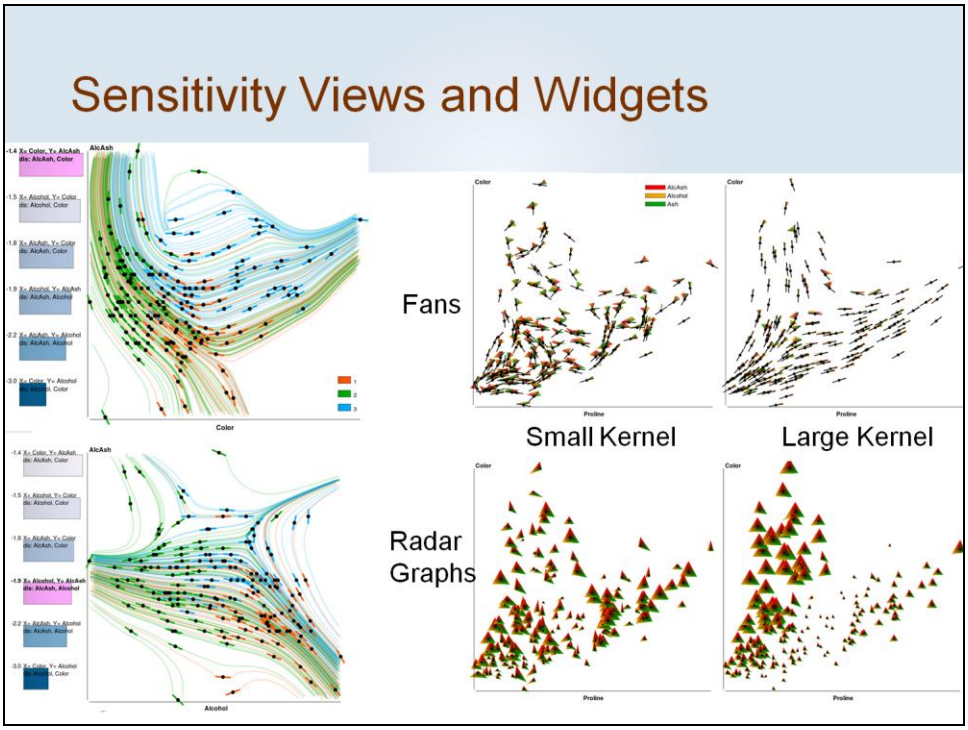
These two example shows that,

In the left plot of **large smoothness (-0.17)**, the streamlines are smooth.

Whereas the plot of **small smoothness** value (-0.18), there are more turns of flows and critical area of mix trends (right and side green nodes have a critical area), such projection are less smooth.

We develop this smoothness metric for projections to have the ranking view.

<next slide>



[LEFT TWO]

By the smoothness estimation by sensitivity, we propose a **Ranking view** to help user navigate different projections efficiently.

Projections are listed **in order of** their smoothness

And by hovering over different bars we can navigate between different projections

We also in ‘terpolate between projections to animate the location change of nodes

so that user can understand different projection.

Also, by this Ranking view interface, users can quickly go over streamlines of projections.

[RIGHT FOUR]

By generalized sensitivity that consider high dimensional space, we also propose sensitivity “Fans” and “Radar Graphs”

Where you can compare sensitivities of (X, Y, Zi) between data quickly, and also compare different third variable Zi used.

In this example in the right we compare three different Z variables.

We also increase kernels to compare the local trend and global trend.

Sensitivity Fans shows the **slope** of the derivatives of (X, Y, Zi), and derivative lines from different Zi variable span a fan on each nodes.

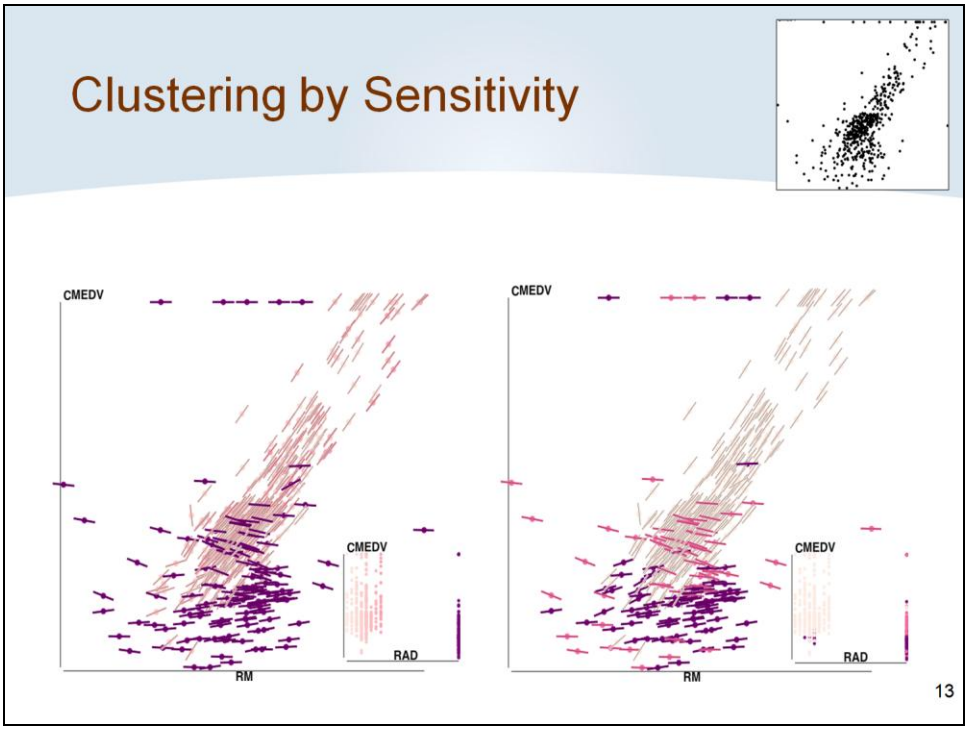
So you can compare and detect anomalies quickly in a local region.

Sensitiviyt Radar Graphs show the **magnitude** of deri of (X, Y, Zi).

The further that vertex of Zi is away from the focus point, the larger the magnitude of the sensitivity is.

In this plot we can quickly find out nodes that are very sensitive to nodes by picking up large spanned triangle.

Also we can see the unbalanced multiple 3D derivatives: equilateral triangles indicate similar sensitivity along different subspace; more striking triangles indicates unbalanced sensitivity.



Sensitivity Scatterplot enable us to do some **non-linear transformations**. Here I will introduce two operations: clustering and selection.

First we can **do clustering by streamlines**.

In the top right we see a scatterplot of the housing price dataset.

Here is another example of clustering by sensitivities.

Sensitivity lines are computed consider X, Y and Z subspace.

We can visually see that nodes are already differentiated by the Z dimension.

This Z dimension has a big gap between two regions, there for it is a good candidate variable to differentiate nodes before projecting to 2D plane.

In the left we see that nodes colored by Z variable (RAD) are differentiated into two different trends:

one positive related (pink) and

one that Y is very insensitive to the change in X (purple).

Then we clustering nodes by these sensitivities of two different trends, as shown in the right.

Clustering results are shown by the color of nodes and lines. There are three clusters: light pink, pink, and purple.

We can see that clustering results also successfully separate nodes of different trends in the purple and the light pink clusters.

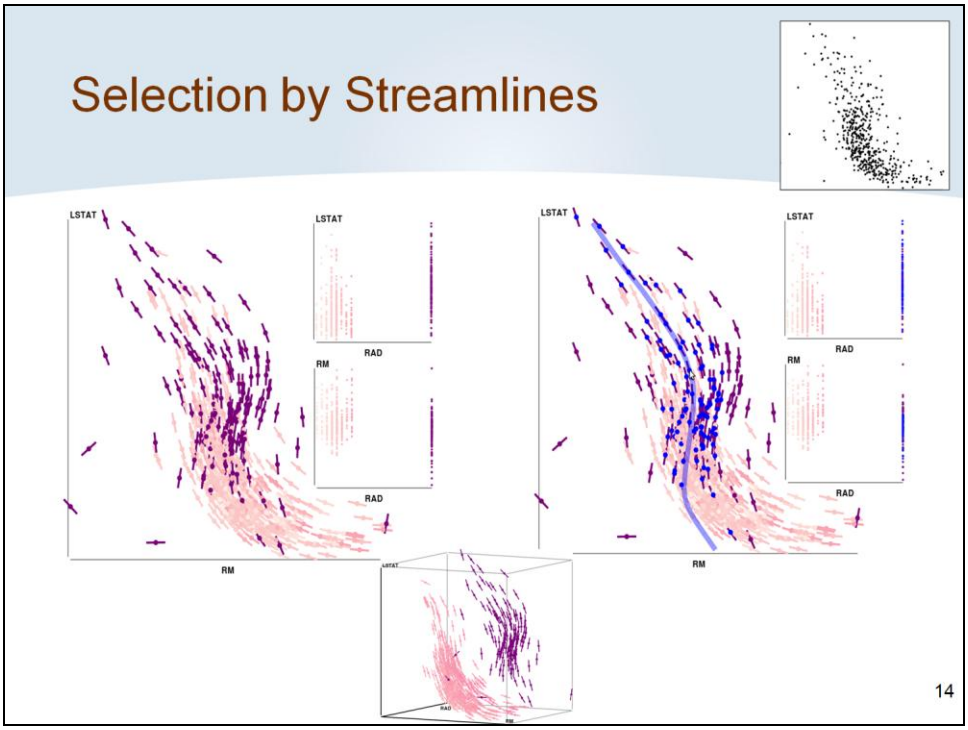
Although the separation is not perfect,

(some nodes with low Z value are miss classified to the purple group with high Z value),

It suggest a grouping of nodes that is only evident in a multi-dimensional space.

This shows that clustering in this subspace (X,Y,Z) may be as accurate as clustering in a possibly high-dimensional space.

We see that clustering based on sensitivities provides a more robust classification of data, since the **streamlines depict the trends** that are **not easy to be captured by linear method**.



Another operation supported by streamlines is **non-linear selection.**

In general we select data points by either axis-aligned selection like drawing a box,
Or drawing a circle,
Or free-form selection that you draw an orbitrary shape with your mouse.
It could be tedious in some cases.

Here we show an example of selection by streamlines.

In this case, sensitivities are approximated considering 3D subspace (X,Y,Z).

We see a separation by coloring nodes by their Z values, the pink group and the purple group.

Now selection by a streamline select nodes surrounding the streamline considering the 3D distance of (X,Y,Z).

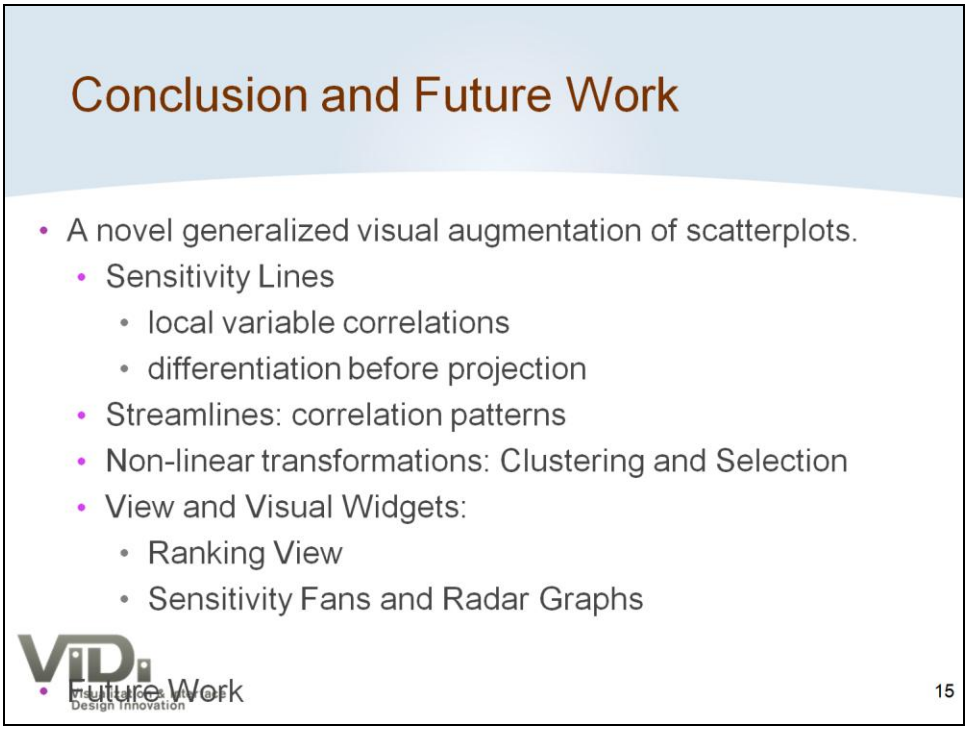
Therefore nodes separated by Z variable would not be selected in this fashion, even thought on 2D plot (X,Y) they are very close.

See the two small scatterplots top right of the right.

Selection by this streamline generated from a purple focus point would only select purple nodes (highlight by navy blue).

No node in the pink group differentiated from the purple group is selected.

Such selection by streamlines are **data-aligned**, **feature-aligned** selection.



I have presented a novel aug of scatterplots By sensitivity lines and streamlines
And some **operations** supported by them such as clustering and selection.
We also proposed the View and the Visual Widgets that help users navigate btw projections efficiently and study relationships between variables.

We provide users a new possible way to explore the data.
to steer visualization, and to navigate data spaces.

In the future we would like to
conduct a thorough **user study** to evaluate
how users can **interpret** generalized sensitivity scatterplots and **draw correct conclusions.**

Also we would like to explore its possibility on **classifications of complex data**
And **parameterized models from machine learning.**

==========

Thank you

- This research was supported in part by the U.S. National Science Foundation through grants CCF-0938114, CCF-0808896, CNS-0716691, and CCF-1025269, the U.S. Department of Energy through the SciDAC program with Agreement No. DE-FC02-06ER25777 and DE-FG02-08ER54956, and HP Labs and AT&T Labs Research.

- chany@cs.ucdavis.edu

- NetZen v1.0 http://vis.cs.ucdavis.edu/software/

ViDi Visualization & Interface Design Innovation