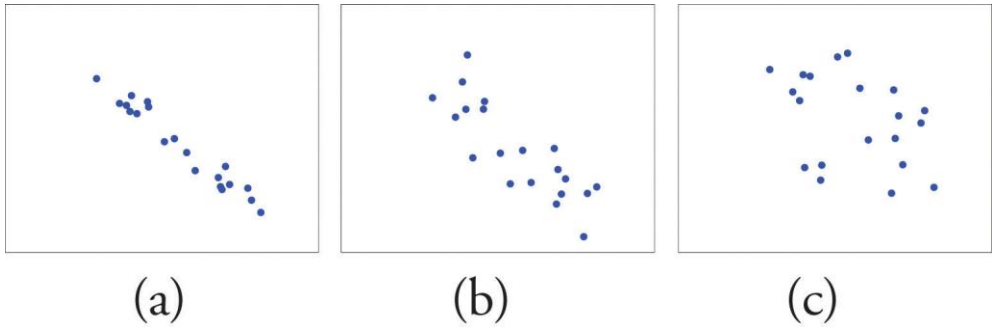
Linear Correlation

Christian Permann

The Problem

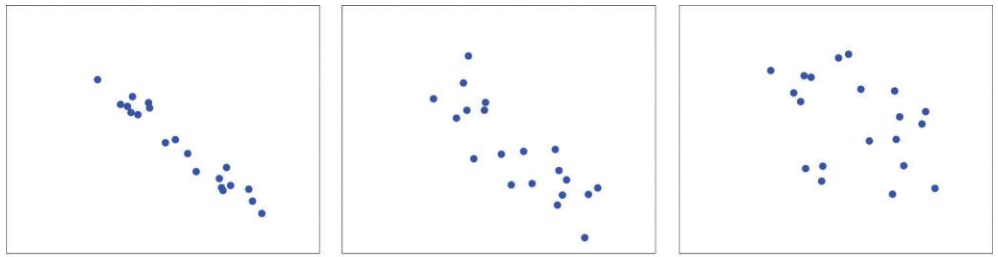
- How well do two variables correlate linearly?



- How can this be quantified?

Image: https://saylordotorg.github.io/text_introductory-statistics/s14-02-the-linear-correlation-coeffic.html

The Method



- Linear Correlation solves this by calculating: $$r = \frac{\Sigma(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\Sigma(x_i - \bar{x})^2 \Sigma(y_i - \bar{y})^2}}$$
- This variable r represents the covariance divided by the product of standard deviations for given variables
- r can take on values from -1 to +1 (including)

Image: https://saylordotorg.github.io/text_introductory-statistics/s14-02-the-linear-correlation-coeffic.html

Maxeler Implementation

- Trivial implementation
- Almost like C/classical Java
- Large performance increase

```
DFEVar n=constant.var(dfeFloat(8, 24), vectorSize);
DFEVar ax=constant.var(dfeFloat(8, 24), 0);
DFEVar ay=constant.var(dfeFloat(8, 24), 0);
DFEVectorType<DFEVar> vectorType = new DFEVectorType<DFEVar>(
    dfeFloat(8,24), vectorSize);

DFEVector<DFEVar> inVector = io.input("inVector", vectorType);
DFEVector<DFEVar> inVector2 = io.input("inVector2", vectorType
    );
for (int i = 0; i < vectorSize; i++){
  ax+=inVector[i];
  ay+=inVector2[i];
}
ax/=n;
ay/=n;

DFEVar xt=constant.var(dfeFloat(8, 24), 0);
DFEVar yt=constant.var(dfeFloat(8, 24), 0);
DFEVar sxx=constant.var(dfeFloat(8, 24), 0);
DFEVar syy=constant.var(dfeFloat(8, 24), 0);
DFEVar sxy=constant.var(dfeFloat(8, 24), 0);
for (int i = 0; i < vectorSize; i++){
  xt = inVector[i]-ax;
  yt = inVector2[i]-ay;
  sxx += xt*xt;
  syy += yt*yt;
  sxy += xt*yt;
}
DFEVar r = sxy/(KernelMath.sqrt(sxx*syy)+1.0e-20);

io.output("outScalar", r, dfeFloat (8, 24));
```

Maxeler Implementation

- To run the program with different array sizes, the following may need to be tweaked:
  - #define Vectors_PCIE_ALIGNMENT (XXX)
    - 16 is default
  - #define Vectors_vectorSize (XXX)
    - 256 is default

Code additions

- If variable length arrays are needed the code needs to be extended
- Such an extension would incur:
  - More complex input
  - More complex read back at different stages
  - Needing more space on the FPGA
  - Possibly using a lot of memory space on the FPGA
  - Possibly slowdowns