# Causal Models for Software Cost Prediction & Control (SCOPE)

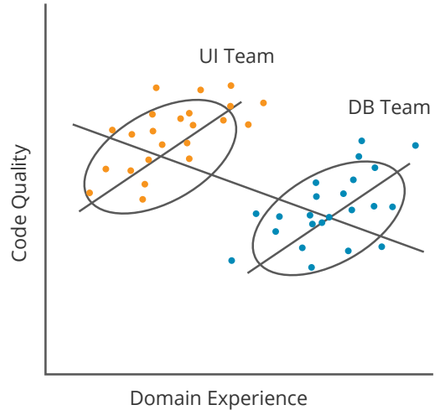
How can we control costs in software development and sustainment? We are collaborating with other researchers to apply causal learning to learn how.

## DoD Problem

- DoD leadership needs to understand why software costs so much.
- DoD program offices need to know where to intervene to control software costs.

## Why Causal Learning?

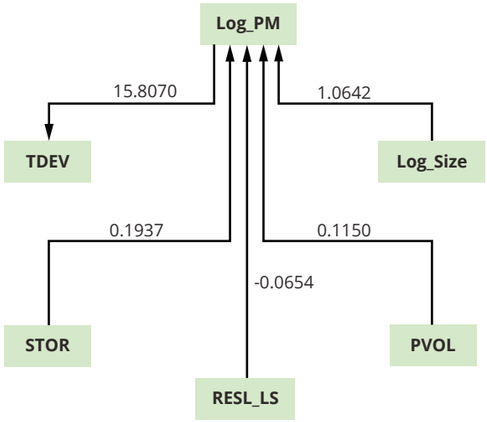
To reduce costs, the causes of an outcome (good or bad) need to be considered. Correlations are insufficient in part due to Simpson's Paradox. For example, in the figure below, if you did not segment your data by team (User Interface [UI] and Database [DB]), you might conclude that increasing domain experience reduces code quality (downward line); however, within each team, it's clear that the opposite is true (two upward lines). Causal learning identifies when factors such as team membership explain away (or mediate) correlations, and it works for much more complicated data sets too.



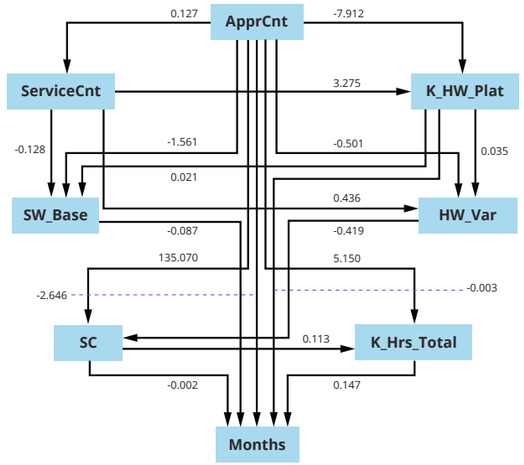

Simpson's Paradox as Applied to UI/DB Data

# **Causal learning** reduces costs.

## Recent Results




COCOMO® II Mini-Cost Estimation Model




Consensus Graph for U.S. Army Software Sustainment

### COCOMO® II – Effort Drivers

Size (SLOC), Team Cohesion, Platform Volatility, Reliability, Storage Constraints, Time Constraints, Product Complexity, Process Maturity, Architecture/Risk Resolution (RESL)

### COCOMO® II – Schedule Drivers

Size (SLOC), Platform Experience, Schedule Constraint, and Effort

### COSYSMO 3.0 – Effort Drivers

Size and Level of Service Requirements

After identifying which of over 40 factors directly drive costs, we used Tetrad to generate mini cost-estimation models that fit well. (In the figure, RESL_LS is the product of RESL and Log_Size.)

A U.S. Army Sustainment data set was segmented into **(Superdomain, ACAT Level)** pairs resulting in five sets of data to search and estimate. Splitting addressed high fan-out for common causes, which can lead to structures typical of Simpson's Paradox. A consensus graph (see above) was built from the resulting five **searched and** fitted models.

For consensus estimation, the data from individual searches was pooled with previously excluded data because of missing values. The resulting 337 releases were used to estimate the consensus graph using Mplus with Bootstrap in estimation.

There was no cherry picking or re-do's—this model is a direct out-of-the-box estimation, achieving good model fit on the first try.

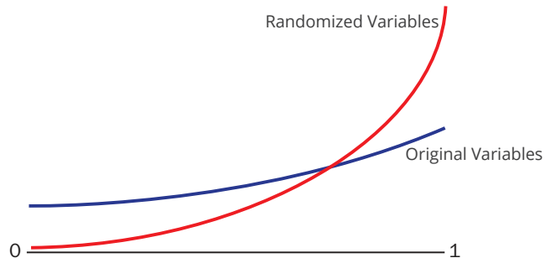

## Acknowledgments

Our thanks to Anandi Hira and Jim Alstad of USC; and Cheryl Jones and her team at U.S. Army AFC-CCDC and DASA-CE.


## Our Solution

Our approach to causal inference is **principled** (i.e., no cherry picking) and **robust** (to outliers). This approach is **especially useful** for small samples—when the number of cases is < 5-10 times the number of variables.

1. Inject **null variables** by appending an independently randomized copy of each original variable.
2. Search (FGES or PC with default settings) with Bootstrap to determine each edge's **Probability of No Edge (PNE)** across the search.
3. Set a **threshold (10th percentile)** among the edges involving a null variable. (Of edges involving a null variable, 90% have a PNE exceeding that threshold.) Then drop the null variables but apply this same threshold to determine which edges to keep among the original variables.




## Summary

Causal models offer better insight for program control than models based on correlation. Knowing which factors drive which program outcomes is essential to sustain the warfighter by providing high-quality, secure software in a timely and affordable manner.

## For More Information

For more information, including causal analyses of other data sets, see our SCOPE Project website.



Mike Konrad, Bob Stoddard, William Nichols, and Dave Zubrow
Michele Falce, Rhonda Brown, and Bryar Wassum

Distribution Statement A: Approved for Public Release;
Distribution is Unlimited

Copyright 2020 Carnegie Mellon University.
This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.
The view, opinions, and/or findings contained in this material are those of the author(s) and should not be construed as an official Government position, policy, or decision, unless designated by other documentation.
References herein to any specific commercial product, process, or service by trade name, trade mark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by Carnegie Mellon University or its Software Engineering Institute.
NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.
[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution. Please see Copyright notice for non-US Government use and distribution.
Internal use:* Permission to reproduce this material and to prepare derivative works from this material for internal use is granted, provided the copyright and "No Warranty" statements are included with all reproductions and derivative works.
External use:* This material may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other external and/or commercial use. Requests for permission should be directed to the Software Engineering Institute at permission@sei.cmu.edu.
* These restrictions do not apply to U.S. government entities.
Personal Software Process℠ and PSP℠ are service marks of Carnegie Mellon University.
DM20-0843