Value Changes Everything

Tim Menzies (tim@menzies.us)
Phillip Green II,
Steve Williams
Oussama Elwaras

Wednesday, May 19, 2009

Sound bites

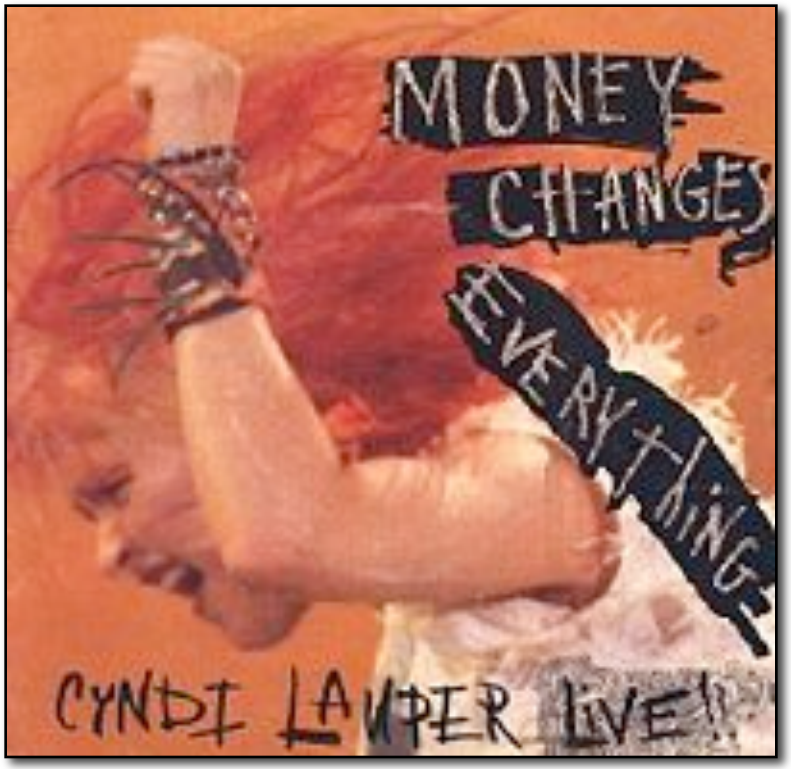
- Value-based SE:
  - not even wrong?
  - Does it change anything?
- Data drought leading to conclusion uncertainty
  - Seek stability over samples
- On sampling some systems, we see
  - Value radically changes the conclusions we reach regarding project organization
- What works best THERE may not work best HERE
  - Needs better ways to find local best

Value-based Software Engineering

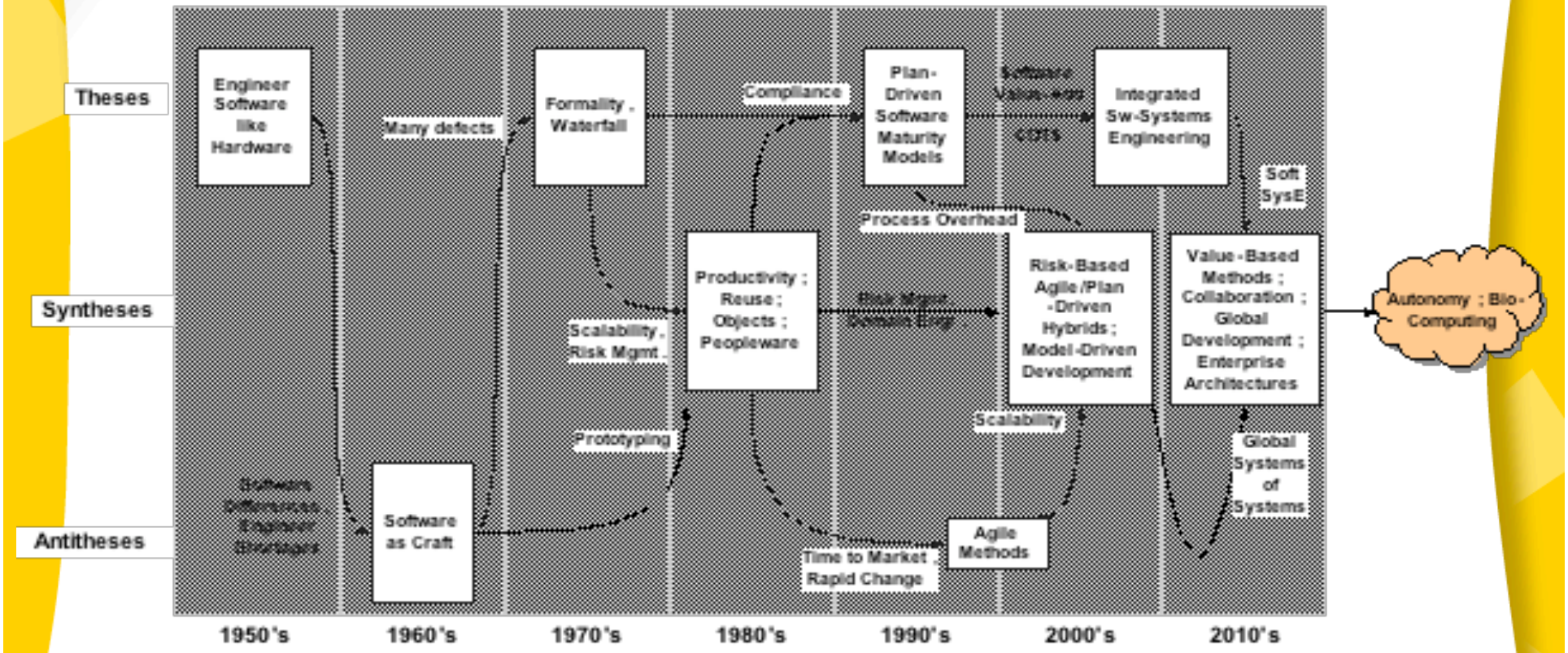
The future of SE?

Thesis: value changes everything!

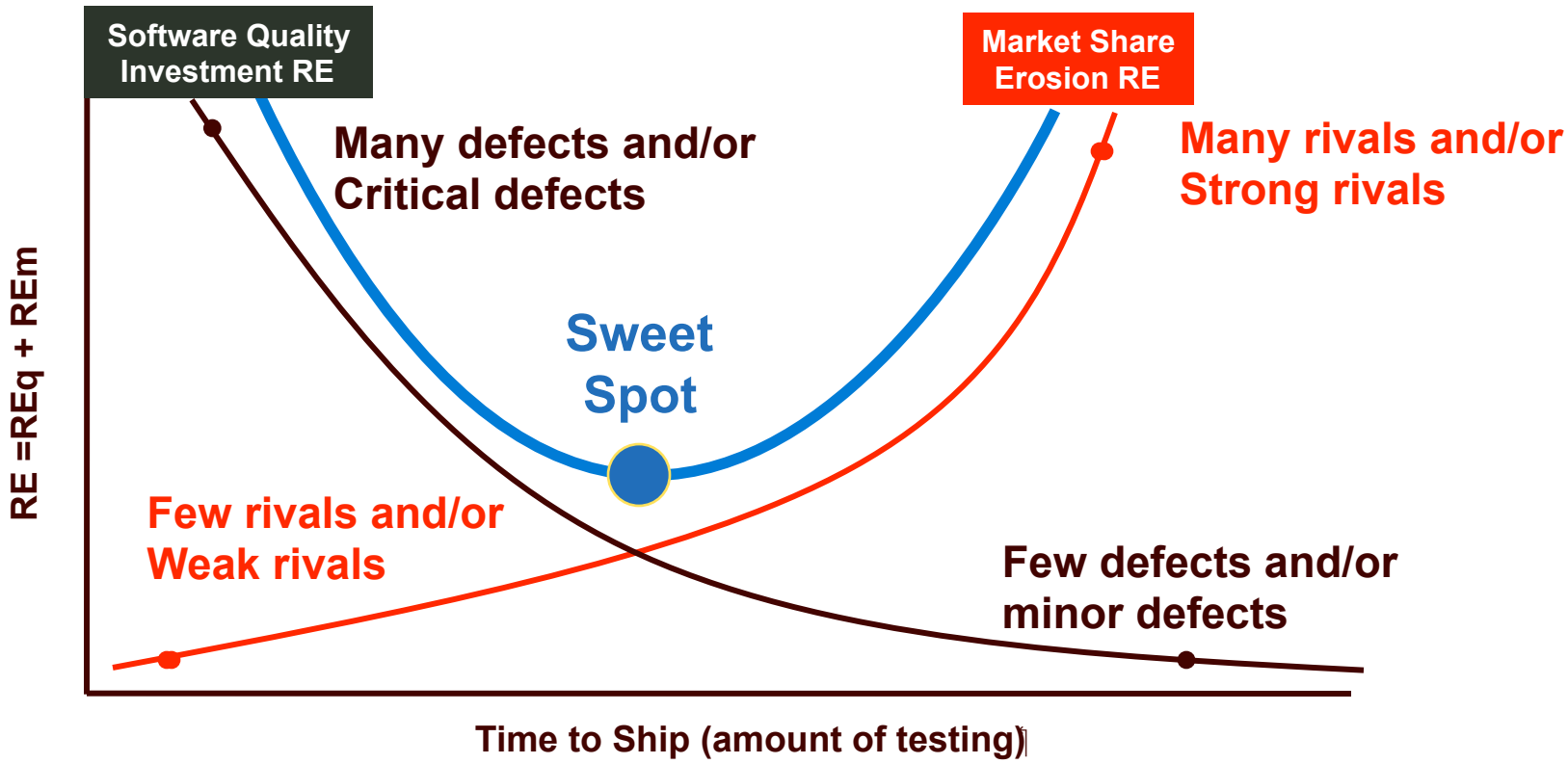
- Q: what is SE
  - A: The application of science and mathematics by which the properties of software are made <u>useful to people</u>
- Most SE techniques are "value-neutral"
  - Boehm, ASE 2004
  - Euphuism for "useless"?
- Value-based SE makes a difference
  - Yeah? Really?



The History of Computing Naturally Leads to Value-based SE



Risk Exposure (RE)
= Software Quality Investment RE (REq)
+ Market Share Erosion RE (REm)



Value-based SE

Not even wrong?

Is the value-thesis not even wrong?

- Wolfgang Pauli
- The "conscience of physics",
  - the critic to whom his colleagues were accountable.
- Scathing in his dismissal of poor theories
  - often labeling it *ganz falsch*, utterly false.
- But *"ganz falsch"* was not his most severe criticism,
  - He hated theories so unclearly presented as to be
    - untestable
    - unevaluatable
  - Worse than wrong because they could not be proven wrong.
  - Not properly belonging within the realm of science
    - even though posing as such.
  - Famously, he wrote of a such unclear paper:
    - "That's not right. It's not *even* wrong."

So is the value thesis refutable?

- Find a domain general “value” proposition
  - Menzies, Boehm, Madachy, Hihn, et al, [ASE 2007]
  - Reduce effort, defects, schedule
  - “energy”
- Find a local value proposition
  - A variant of USC Ph.D. thesis
    - [Huang 2006]: Software Quality Analysis: a Value-Based Approach
  - “value”
- Use them in a what-if scenario
- Any difference in the conclusions?

```
(defun energy ()
  "Calculates energy based on cocomo pm, tdev, coqualmo defects,
Madachy's risk."
  (let* ((npm   (calc-normalized-pm))
         (ntdev   (calc-normalized-tdev))
         (ndefects  (calc-normalized-defects))
         (nrisk   (calc-normalized-risk))
         (pm-weight 1)
         (tdev-weight 1)
         (defects-weight (+ 1 (expt 1.8 (- (xomo-rating? 'rely) 3))))
         (risk-weight 1))
    (/ (sqrt (+ (expt (* npm  pm-weight) 2)
                (expt (* ntdev tdev-weight) 2)
                (expt (* ndefects defects-weight) 2)
                (expt (* nrisk risk-weight) 2)))
       (sqrt (+ pm-weight tdev-weight
                defects-weight risk-weight)))))
```

```
(defun risk-exposure ()
  “Calculates risk exposure based on rely”
  (let* ((pm (calc-pm))
         (size-coefficient (calc-size-coefficient '(rely)))
         (defects (calc-defects))
         (defects_vl (calc-defects-with-vl-rely))
         (loss-probability (/ defects defects_vl))
         (loss-size (* (expt 3 (/ (- (xomo-rating? 'cplx) 3) 2) )
                       size-coefficient
                       pm))
         (software-quality-re (* loss-probability loss-size))
         (market-coefficient (calc-market-coefficient '(rely)))
         (market-erosion-re (* market-coefficient pm))
    (+ software-quality-investment-re
       market-erosion-re)))
```

Aside

- Not really [Huang06]
  - But some variant Huang06
- Had to use some “engineering judgment”
  - a.k.a. guesses
- Apologies to Dr. Huang

Tools

- Four USC models
  - COCOMO effort prediction: staff months
  - COCOMO schedule predictor: calendar months
  - COQUALMO defect predictor: defects/KLOC
- Monte Carlo simulator
- AI search engine
  - Search for the least number of project changes …
  - … that most improves the “target”
  - “Target” is either
    - [Ase07]’s “energy” function
    - [Huang06]’s “XPOS” proposition (risk exposure)

“Energy” [Ase07]

- Euclidean Distance to lowest everything

- Energy = sqrt(a * square(normalized (Time)) +
  b * square(normalized(Effort)) +
  c * square(normalized(Defects))
)
  / sqrt(a+b+c)

Xpos [Huang06]

- Value based evaluation method designed to minimize risk exposure based on 'rely'
- Balances beating everyone to market with more/worse bugs and being last to market with few/minor bugs.
- Based on NASA/USC Inspector SCRover project described in [Huang06]
  - XPOS
  - Risk Exposure (RE)
    = Software Quality Investment RE (REq)
    + Market Share Erosion RE (REm)

Software Quality Investment = Pq(L) * Sq(L)

- Pq(L):
  - [Huang06] calculated from COQUALMO estimates of delivered defect density
  - To incorporate COQUALMO model: defects/defects-with-vl-rely
- Sq(L)
  - [Huang06] used based values from a Pareto distribution and modified it with a coefficient based on a factor depending if a project was for early startup (1/3) commercial (1), high finance(3)
  - We used the same values for the distribution but instead of defining 3 different functions, we used a function base don cplx to determine the coefficient 3^((cplx-3)/2)) (range is [0.3333 .. 5.196]

Market Share Erosion Risk Exposure (REm)

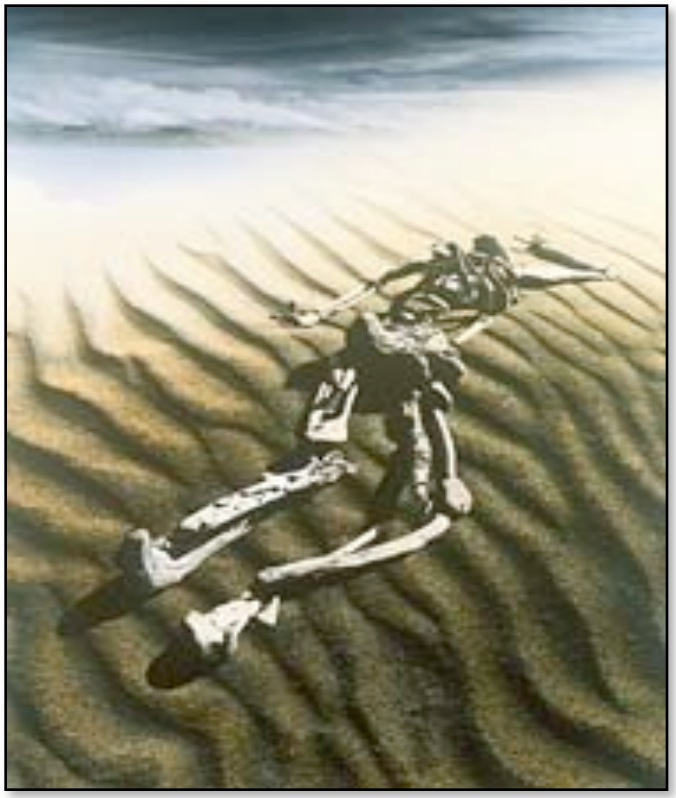
- [Huang06]
  - used a simple exponential distribution for Rem
  - REm was normalized
- We weight it with PM

The details

Using AI to find stable conclusions
in a space of options

Problem: local tuning

- Problem
  - Models need calibration
  - Calibration needs data
  - Usually, data incomplete (the “data drought”)
- Our thesis :
  - Precise tunings not required
  - Space of possible tunings is well-defined
  - Find and set the collars
    - Reveal policies that reduce effort/ defects months
    - That are stable across the entire space



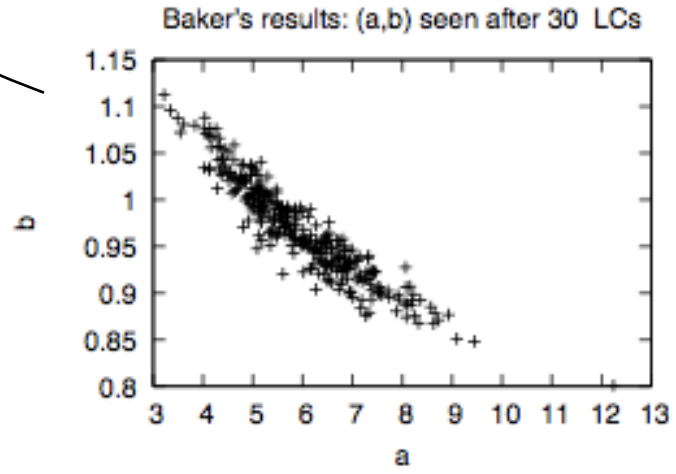
Experts disagree

No majority view

- Target application picked
  - A mission critical, real-time system;
  - Built by contractors (not in-house)
  - That has an operational life of 5 to 10 years (since have invested much effort into a mission critical system, an organization is most likely to use it for many years to come).
- For each COCOMO input variable
  - Boehm defines each variable
  - 5 minutes “open comments”
  - Vote. Record majority view

The tuning instability problem

If some method DOUBLED productivity, you might miss it if tunings randomly jumps 9 to 4.5.



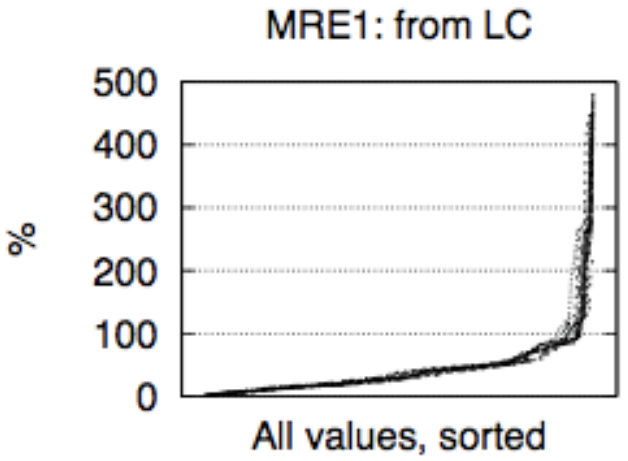


Fig. 5. Results of applying LC numerous times to 90% of the NASA93 data sets (available from `http://promisedata.org/data`). Left-hand-side shows computed (a, b) values. Right-hand-side shows MRE1s generated over the NASA93 data set for ten case studies (one study per line).

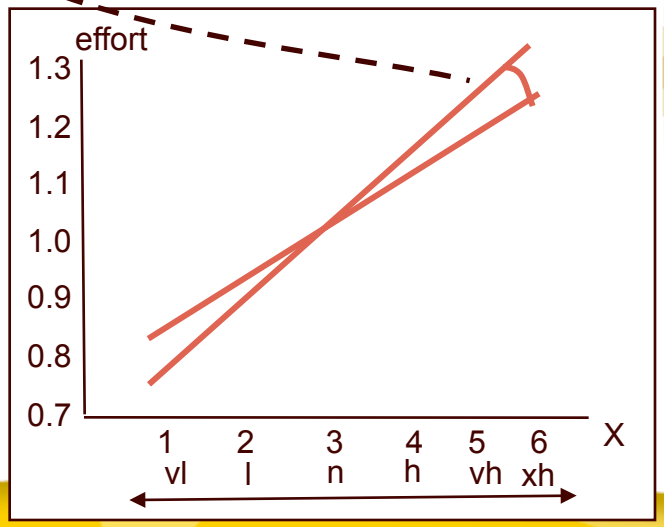
Dodging tuning instability

- Estimate = model(project, tunings)
- Twiddle project
- Let tunings roam free
- Can still control the estimate (if project dominates estimate)
- Project details are the dominate influence on estimate for the USC models.

Sampling

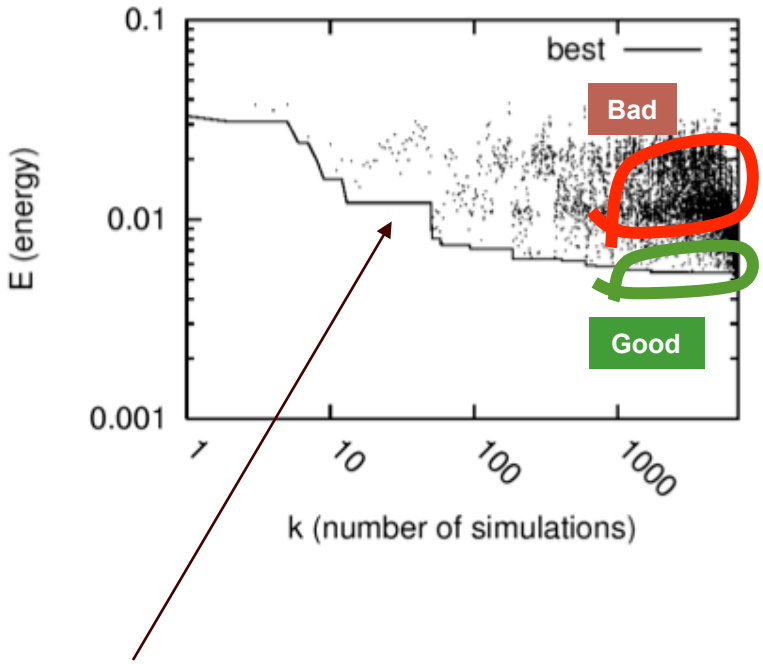
- E.g. effort = mx + b
- Two kinds of unknowns
  - Unknowns in project ranges
    - E.g. range of “x”
  - Unknowns in internal ranges
    - E.g. range of {“m”, “b”}
- Standard practice:
  - Use historical data to constrain {“m”,”b”}
- Here: Monte Carlo over range of { “x” , “m”, “b” }
  - Learn values for “x” that reduce effort
  - As a side-effect, reduce variance
  - Not need for tuning data

	ranges			values	
project	feature	low	high	feature	setting
Flight:	rely	3	5	tool	2
	data	2	3	sced	3
	cplx	3	6		
	time	3	4		
	stor	3	4		
	pvol	2	4		
	acap	3	5		
	apex	2	5		
	pcap	3	5		
	plex	1	4		
	ltex	1	4		
	pmat	2	3		
	Ksloc	7	418		
Ground:	rely	1	4	tool	2
	data	2	3	sced	3
	cplx	1	4		
	time	3	4		
	stor	3	4		
	pvol	2	4		
	acap	3	5		
	apex	3	5		
	pcap	3	5		
	plex	1	4		
	ltex	1	4		
	pmat	2	3		
	Ksloc	11	392		



Search for stable conclusions

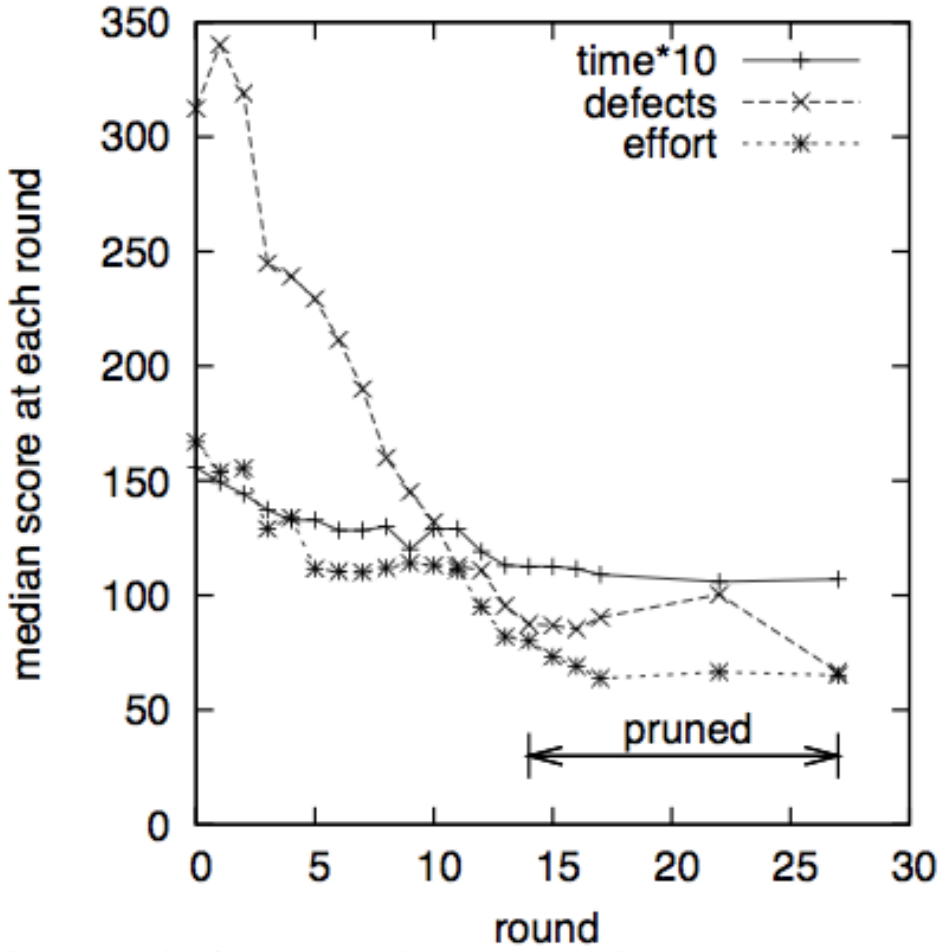
- Using simulated annealing, Monte Carlo simulated annealing across intersection of
  - A particular project type
  - Space of possible tunings
- Rank options by frequency in **good**, not **bad**
- For r options
  - Try setting the $1 \le x \le R$ top ranked options
  - Simulate (100 times) to check the effect of options 1 .. x
- Smile if
  - Reduced median and variance in defects/ efforts/ time/ threats



Sample run
(after 10,000 runs, little improvement)

Other search methods

- A-star
- MaxWalkSat
- Isamp
- Etc



Decisions made from round=1 to round=13:

round0: $r_x = \emptyset$	round7: added {rely=3}
round1 added {pmat=3}	round8: added {stor = 3}
round2: added {resl=4}	round9: added {time = 3}
round3: added {team=5}	round10: added {tool = 4}
round4: added {aexp=4}	round11: added {sced = 2}
round5: added {docu=3}	round12: added {site = 4}
round6: added {plex=4}	round13: added {acap = 5}

What works best?

- A little domain knowledge goes a long way
- Standard methods not best
- Best methods very effective

algorithm	Defects	months	time
SEESAW	4	4	3
BEAM	0	3	3
A-star	0	1	1
SA	0	1	1
MaxWalkSat	0	0	0
ISSAMP	0	0	0

data set	defects	time	effort
flight	80%	39%	72%
ground	85%	38%	73%
osp	65%	4%	42%
ops2	26%	22%	5%
median	73%	30%	57%

Figure 6: Percent reductions $(1 - final/initial)$ achieved by SEESAW on the Figure 3 case studies. The *initial* values come from round 0 of the forward select. The *final* values come from the policy point. Note that all the initial and final values are statistically different (Mann-Whitney, 95% confidence).

Results

And the winner is…

…. no one in particular

Data	Range	*value* B=BFC	*value* X=XPOS	$\frac{B}{B+X}$	defect removal manual	defect removal automatic
ground	rely = 4	70	20	77		
	aa = 6	70	25	73		hi in B
	resl = 6	65	40	61		
	etat = 1	35	65	35		lo in X
	aexp = 5	45	85	34		
	pr = 1	35	80	30	lo in X	
	aa = 1	25	60	29		lo in X
	data = 2	25	70	26		
	rely = 1	15	70	17		
flight	rely = 5	65	25	72		
	flex = 6	80	50	61		
	docu = 1	55	85	39		
	site = 6	55	85	39		
	resl = 6	45	70	39		
	pr = 1	45	70	39	lo in X	
	pvol = 2	45	75	37		
	data = 2	35	60	36		
	cplx = 3	45	90	33		
	rely = 3	15	60	20		
OSP	pmat = 4	85	45	65		
	resl = 3	45	70	39		
	ruse = 2	40	65	38		
	docu = 2	25	90	21		
OSP2	sced = 2	100	0	100		
	sced = 4	0	80	0		

Figure 7: Frequency (in percents) of feature ranges seen in 20 repeats of SEESAW, using two different goal functions: BFC and XPOS. The last two columns comment on any defect reduction feature. Not shown in this figure are any feature ranges that occur less than 50% of the time.

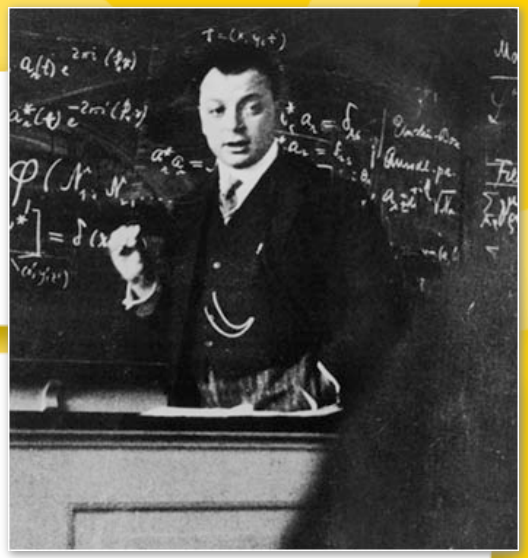
aa 1
aexp 1
cplx 1
data 2
docu 1
etat 1
flex 1
pmat 1
Pr 2
pvol 1
ruse 1
rely 2
resl 2
sced 1
site 1

MIA: prec, team, acap, ltex, pcap, pcon,stor, time,tool

Conclusion

So what?

Conclusion



- Is value-based SE “ganz falsch”? (not even wrong)
  - Hard to tell, if we have a data drought
  - So seek stability in samples of the possibilities
- On sample, using 4 case studies and 2 value functions:
  - Many seemingly important factors weren’t (important)
  - The most important ones change from project to project
  - For any project, changes to value changes everything

Conclusions

- Value-based SE:
  - Not “not even wrong”
  - Value change everything
- Data drought leading to conclusion uncertainty
  - Seek stability over samples
- On sampling some systems, we see
  - Value radically changes the conclusions we reach regarding project organization
- What works best THERE may not work best HERE
  - Needs better ways to find local best