XOMO: understanding Development Options for Autonomy

tim@menzies.us
PDX, USA

Julian Richardson
julianr@email.arc.nasa.gov
RIACS,USRA

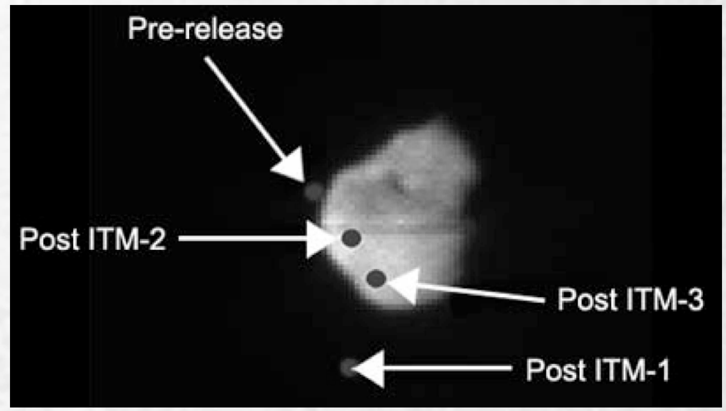
Sound bites

- 21st software: uses autonomy. Are you ready?
- AI software is still software
- SE has much to offer AI
- AI has much to offer SE
- If you can’t mine data, grow it.
- Monte Carlo + data miners = good
- Conclusions without local calibration

Autonomy: example

Good news: Autonomy reduces flight risks!

- **Deep Impact intercepted comet Tempel 1, July 4th 2005.**
- **On-board autonomy made three last-minute trajectory corrections.**
  1. T-minus 90 minutes.
  2. T-minus 35 minutes
  3. T-minus 12.5 minutes
- **Note: ground control could not have made the last correction**
  - Asteroid was 7½ light minutes from earth; i.e., 15 minutes round trip;
  - "You can't joystick this thing."
    -- Deep Impact mission controller



http://www.nasa.gov/mission_pages/deepimpact/multimedia/SHYAM.html

- Risks mitigated:
  1. failure of ground-commanded trajectory calculations (based on old data, may be slow)
  2. failure due to communications outage

Challenge: How to build intelligent systems in a cost-effective manner?

Why autonomy?

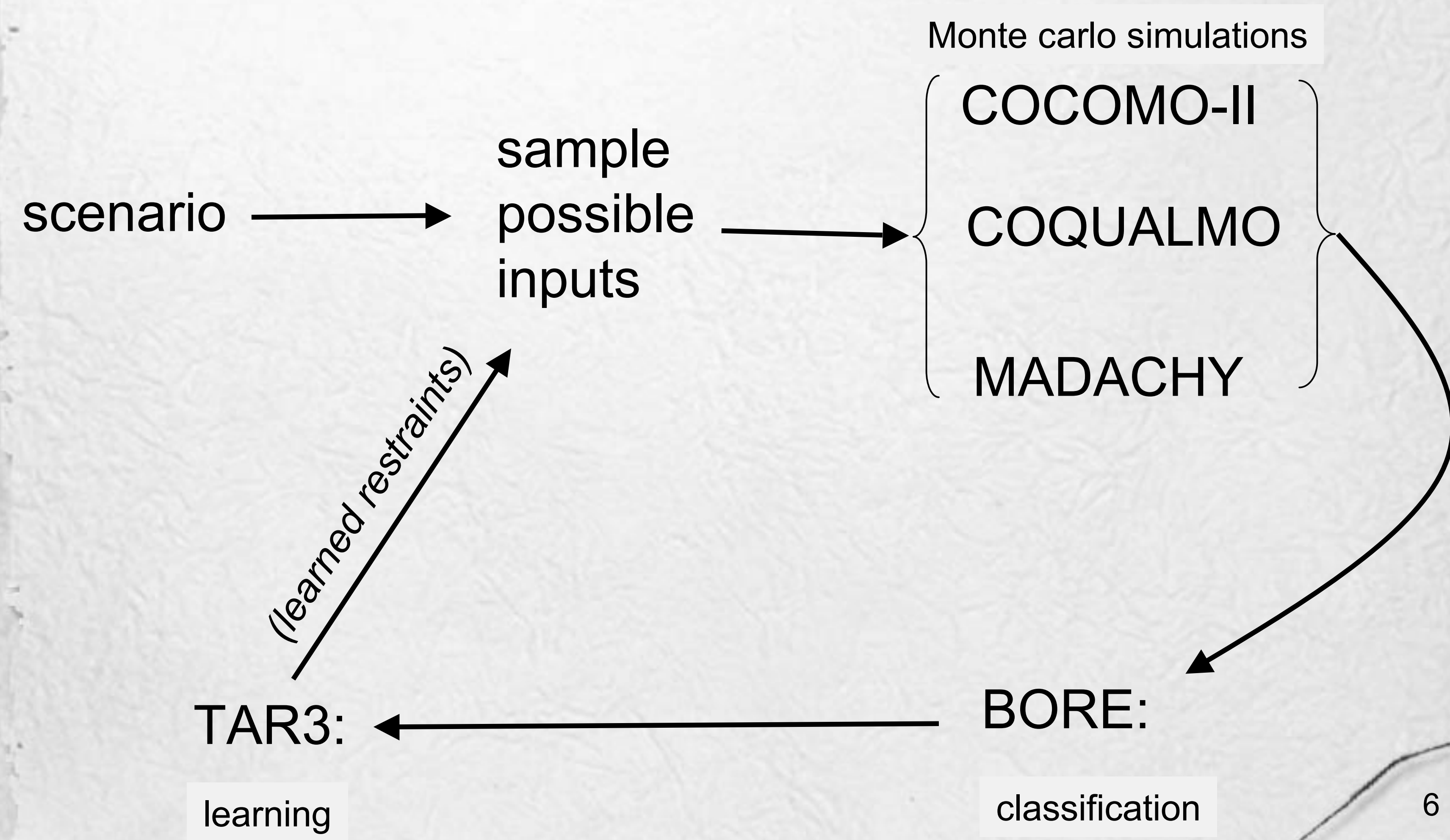
- Extends capabilities:
  - Automatic rendezvous and docking
    - Good for in-orbit assembly
  - Faster reaction to science event
    - Data collection > downlink capacity
  - Extended mission life
    - Less reliance on ground control
      - Saves time (few controllers)
      - Avoids human errors (e.g XXXX)
    - Historical data: 41% of software anomalies triggered by communications uplink/downlink [Lutz 2003]

Scenario

- Talented PhD-level programmers
- No prior autonomy experience
- High reliability
- Complex software
- Hope that product will be reusable

- Ksloc = 75 .. 125
- Rely = 5
- Prec = 1
- Acap = 6
- Aexp = 1
- Cplx = 6
- Ltex = 1
- Reus = 6
- Pmat, time, resl, …
  - = ?

Cycle: repeat till happy (or no more improvement)



COCOMO models

COCOMO-II effort model

	vl	l	n	h	vh	xh
Scale factors:						
flex	5.07	4.05	3.04	2.03	1.01	
pmat	7.80	6.24	4.68	3.12	1.56	
prec	6.20	4.96	3.72	2.48	1.24	
resl	7.07	5.65	4.24	2.83	1.41	
team	5.48	4.38	3.29	2.19	1.01	
Effort multipliers:						
acap	1.42	1.19	1.00	0.85	0.71	
aexp	1.22	1.10	1.00	0.88	0.81	
cplx	0.73	0.87	1.00	1.17	1.34	1.74
data		0.90	1.00	1.14	1.28	
docu	0.81	0.91	1.00	1.11	1.23	
ltex	1.20	1.09	1.00	0.91	0.84	
pcap	1.34	1.15	1.00	0.88	0.76	
pcon	1.29	1.12	1.00	0.90	0.81	
plex	1.19	1.09	1.00	0.91	0.85	
pvol		0.87	1.00	1.15	1.30	
rely	0.82	0.92	1.00	1.10	1.26	
ruse		0.95	1.00	1.07	1.15	1.24
sced	1.43	1.14	1.00	1.00	1.00	
site	1.22	1.09	1.00	0.93	0.86	0.80
stor			1.00	1.05	1.17	1.46
time			1.00	1.11	1.29	1.63
tool	1.17	1.09	1.00	0.90	0.78	

$$months = a * \left(KSLOC^{\left(b+0.01*\sum_{i=1}^{5} SF_i\right)}\right) * \left(\prod_{j=1}^{17} EM_j\right) \quad (1)$$

Madachy Risk Model: how many dumb things are you doing today?

		vl	l	n	h	vh	xh
	rely						
sced	vl				1	2	
	l					1	
	cplx						
sced	vl				1	2	4
	l					1	2
	n						1
	time						
sced	vl				1	2	4
	l					1	2
	n						1
	pvol						
sced	vl				1	2	
	l					1	
	tool						
sced	vl	2	1				
	l	1					
	pexp						
sced	vl	4	2	1			
	l	2	1				
	n	1					
	pcap						
sced	vl	4	2	1			
	l	2	1				
	n	1					
	aexp						
sced	vl	4	2	1			
	l	2	1				
	n	1					
	acap						
sced	vl	4	2	1			
	l	2	1				
	n	1					
	ltex						
sced	vl	2	1				
	l	1					
	pmat						
sced	vl	2	1				
	l	1					

		vl	l	n
	acap			
rely	n	1		
	h	2	1	
	vh	4	2	1
	pcap			
rely	n	1		
	h	2	1	
	vh	4	2	1
	acap			
cplx	h	1		
	vh	2	1	
	xh	4	2	1
	pcap			
cplx	h	1		
	vh	2	1	
	xh	4	2	1
	tool			
cplx	h	1		
	vh	2	1	
	xh	4	2	1
	pmat			
rely	n	1		
	h	2	1	
	vh	4	2	1
	acap			
pmat	vl	2	1	
	l	1		
	acap			
stor	h	1		
	vh	2	1	
	xh	4	2	1
	acap			
time	h	1		
	vh	2	1	
	xh	4	2	1
	acap			
tool	vl	2	1	
	l	1		
	pcap			
tool	vl	2	1	
	l	1		

		vl	l	n
	aexp			
ruse	h	1		
	vh	2	1	
	xh	4	2	1
	ltex			
ruse	h	1		
	vh	2	1	
	xh	4	2	1
	pcap			
pmat	vl	2	1	
	l	1		
	pcap			
stor	h	1		
	vh	2	1	
	xh	4	2	1
	pcap			
time	h	1		
	vh	2	1	
	xh	4	2	1
	pcap			
ltex	vl	4	2	1
	l	2	1	
	n	1		
	pexp			
pvol	h	1		
	vh	2	1	
	pmat			
tool	vl	2	1	
	l	1		
	tool			
time	vh	1		
	xh	2	1	
	aexp			
team	vl	2	1	
	l	1		
	sced			
team	vl	2	1	
	l	1		
	site			
team	vl	2	1	
	l	1		

COQUALMO: defect introduction

	rely	data	ruse	docu	cplx	time	stor	pvol	acap	pcap	pcon	aexp	plex	ltex	tool	site	sced
requirements:																	
xh			1.05		1.32	1.08	1.08	1.16								0.83	
vh	0.7	1.07	1.03	0.86	1.21	1.05	1.05	1.1	0.75	1	0.82	0.81	0.9	0.93	0.92	0.89	0.85
h	0.85	1.04	1.02	0.93	1.1	1.03	1.03	1.05	0.87	1	0.91	0.91	0.95	0.97	0.96	0.95	0.92
n	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
l	1.22	0.93	0.95	1.08	0.88			0.86	1.17	1	1.11	1.12	1.05	1.04	1.05	1.1	1.09
vl	1.43			1.16	0.76				1.33	1	1.22	1.24	1.11	1.07	1.09	1.2	1.18
design:																	
xh			1.02		1.41	1.2	1.18	1.2									
vh	0.69	1.1	1.01	0.85	1.27	1.13	1.12	1.13	0.83	0.85	0.8	0.82	0.86	0.88	0.91		
h	0.85	1.05	1	0.93	1.13	1.06	1.06	1.06	0.91	0.93	0.9	0.91	0.93	0.91	0.96		
n	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1		
l	1.23	0.91	0.98	1.09	0.86			0.83	1.1	1.09	1.13	1.11	1.09	1.07	1.05		
vl	1.45			1.18	0.71				1.2	1.17	1.25	1.22	1.17	1.13	1.1		
coding:																	
xh			1.02		1.41	1.2	1.15	1.22									
vh	0.69	1.1	1.01	0.85	1.27	1.13	1.1	1.15	0.9	0.76	0.77	0.88	0.86	0.82	0.8		
h	0.85	1.05	1	0.92	1.13	1.06	1.05	1.08	0.95	0.88	0.88	0.94	0.94	0.91	0.9		
n	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1		
l	1.23	0.91	0.98	1.09	0.86			0.82	1.05	1.16	1.15	1.07	1.08	1.11	1.13		
vl	1.45			1.18	0.71				1.11	1.32	1.3	1.13	1.16	1.22	1.25		

	prec	flex	resl	team	pmat
requirements:					
xh	0.7	1	0.76	0.75	0.73
vh	0.84	1	0.87	0.87	0.85
h	0.92	1	0.94	0.94	0.93
n	1	1	1	1	1
l	1.22	1	1.16	1.17	1.19
vl	1.43	1	1.32	1.34	1.38
design:					
xh	0.75	1	0.7	0.8	0.61
vh	0.87	1	0.84	0.9	0.78
h	0.94	1	0.92	0.95	0.89
n	1	1	1	1	1
l	1.17	1	1.22	1.13	1.33
vl	1.34	1	1.43	1.26	1.65
coding:					
xh	0.81	1	0.71	0.86	0.63
vh	0.9	1	0.84	0.92	0.79
h	0.95	1	0.92	0.96	0.9
n	1	1	1	1	1
l	1.12	1	1.21	1.09	1.3
vl	1.24	1	1.41	1.18	1.58

COQUALMO: defect removal

	automated analysis	peer reviews	execution_testing _and_tools
requirements:			
xh	0.4	0.7	0.6
vh	0.34	0.58	0.57
h	0.27	0.5	0.5
n	0.1	0.4	0.4
l	0	0.25	0.23
vl	0	0	0
design:			
xh	0.5	0.78	0.7
vh	0.44	0.7	0.65
h	0.28	0.54	0.54
n	0.13	0.4	0.43
l	0	0.28	0.23
vl	0	0	0
coding:			
xh	0.55	0.83	0.88
vh	0.48	0.73	0.78
h	0.3	0.6	0.69
n	0.2	0.48	0.58
l	0.1	0.3	0.38
vl	0	0	0

XOMO=
support code for COCOMO Monte Carlos

```
# thousands of lines of codes
_ANY(ksloc, 2, 10000)

# scale factors: exponential effect on effort
ANYi(prec, 1, 6)
ANYi(flex, 1, 6)
...
# effort multipliers: linear effect on effort
ANYi(rely, 1, 5)
...
# defect removal methods
_ANYi(automated_analysis, 1, 6)
_ANYi(peer_reviews, 1, 6)
_ANYi(execution_testing_and_tools, 1, 6)
…
# calibration parameters
_ANY(a, 2.25,3.25)
_ANY(b, 0.9, 1.1)
```

```
function Prec()
   return scaleFactor("prec", prec())
...
function Effort() {
      return A() * Ksloc() ˆ E() *
      Rely()* Data()* Cplx()*
      Ruse()* Docu()* Time()* Stor()* Pvol()*
      Acap()*Pcap()* Pcon()* Aexp()* Plex()*
      Ltex()* Tool()*Site()* Sced() }


function E() {
      return B() +
       0.01*(Prec() + Flex()
            + Resl() + Team() + Pmat())}
```

Case study

Sample call

```
runxomo() {
  Scenario="-p ksloc -l 75 -u 125
            -p rely -= 5
            -p prec -= 1
            -p acap -= 5
            -p aexp -= 1
            -p cplx -= 6
            -p ltex -= 1
            -p ruse -= 6"
  xomo $Scenario }
```

Sample output

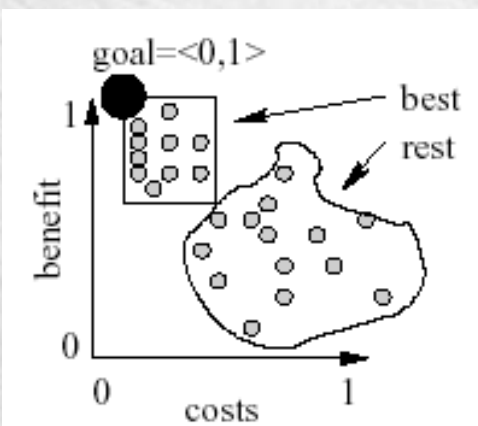
26 inputs								*3 outputs*		
rely	*plex*	*ksloc*	…	*pcap*	*time*	*aa*		*effort*	*schedule risk*	*defects*
5	1	118.80	…	5	3	5		2083	69	0.50
5	1	105.51	…	1	3	5		4441	326	0.86
5	4	89.26	…	3	5	3		1242	63	0.96
5	2	89.66	…	1	4	5		2118	133	2.30
5	1	105.45	…	2	4	5		6362	170	2.66
5	3	118.43	…	2	6	2		7813	112	4.85
5	4	110.84	…	4	4	4		4449	112	6.81
…										

Effort does not predict for defect density

Highest schedule risk, one of the lowest defects

BORE: best or rest selection

- Binary classification of N-utilities
  - Effort
  - Defects
  - Schedule risk

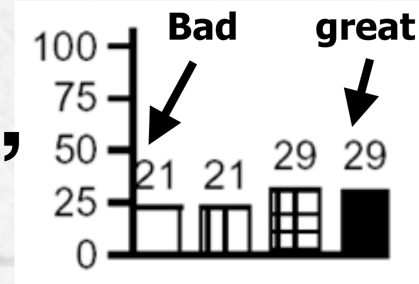


26 inputs							3 outputs		
rely	*plex*	*ksloc*	…	*pcap*	*time*	*aa*	*effort*	*schedule risk*	*defects*
5	1	118.80	…	5	3	5	2083	69	0.50
5	1	105.51	…	1	3	5	4441	326	0.86
5	4	89.26	…	3	5	3	1242	63	0.96
5	2	89.66	…	1	4	5	2118	133	2.30
5	1	105.45	…	2	4	5	6362	170	2.66
5	3	118.43	…	2	6	2	7813	112	4.85
5	4	110.84	…	4	4	4	4449	112	6.81
…									

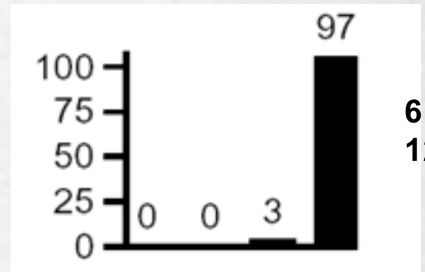
	rely	*plex*	*ksloc*	…	*pcap*	*time*	*aa*	*effort*	*secdRisk*	*defects*
best:										
	5	4	89.26	…	3	5	3	1242	63	0.96
	5	1	118.80	…	5	3	5	2083	69	0.50
	5	2	89.66	…	1	4	5	2118	133	2.30
rest:										
	5	1	105.51	…	1	3	5	4441	326	0.86
	5	4	110.84	…	4	4	4	4449	112	6.81
	5	3	118.43	…	2	6	2	7813	112	4.85

The TAR3 "treatment learner"

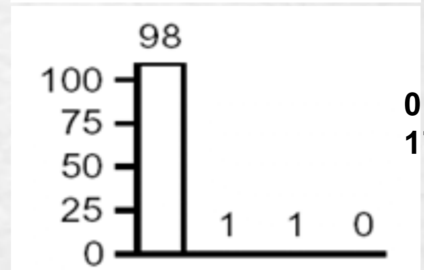
- **Classes have utilities (best > rest)**
- **"treatment"= policy**
  - **what to do**
  - **what to watch for**
- **seek attribute ranges that are**
  - **often seen in "good"**
  - **rarely seen in "bad".**
- **Treatment=**
  - **constraint that changes baseline frequencies**



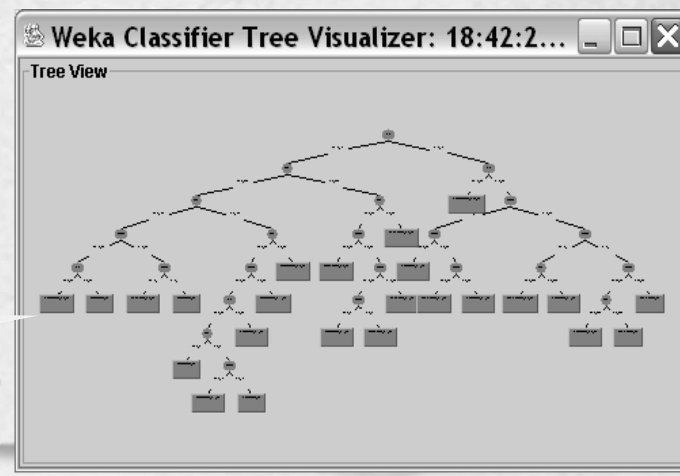
Housing, baseline (% of housing types)



6.7 <= rm < 9.8 and 12.6 <= ptratio < 15.9



0.6 <= NOX < 1.9 and 17.16 <= lstat < 39

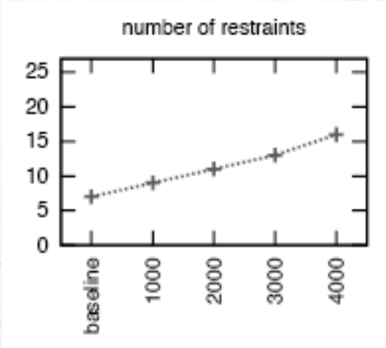


A few variables are (often) enough

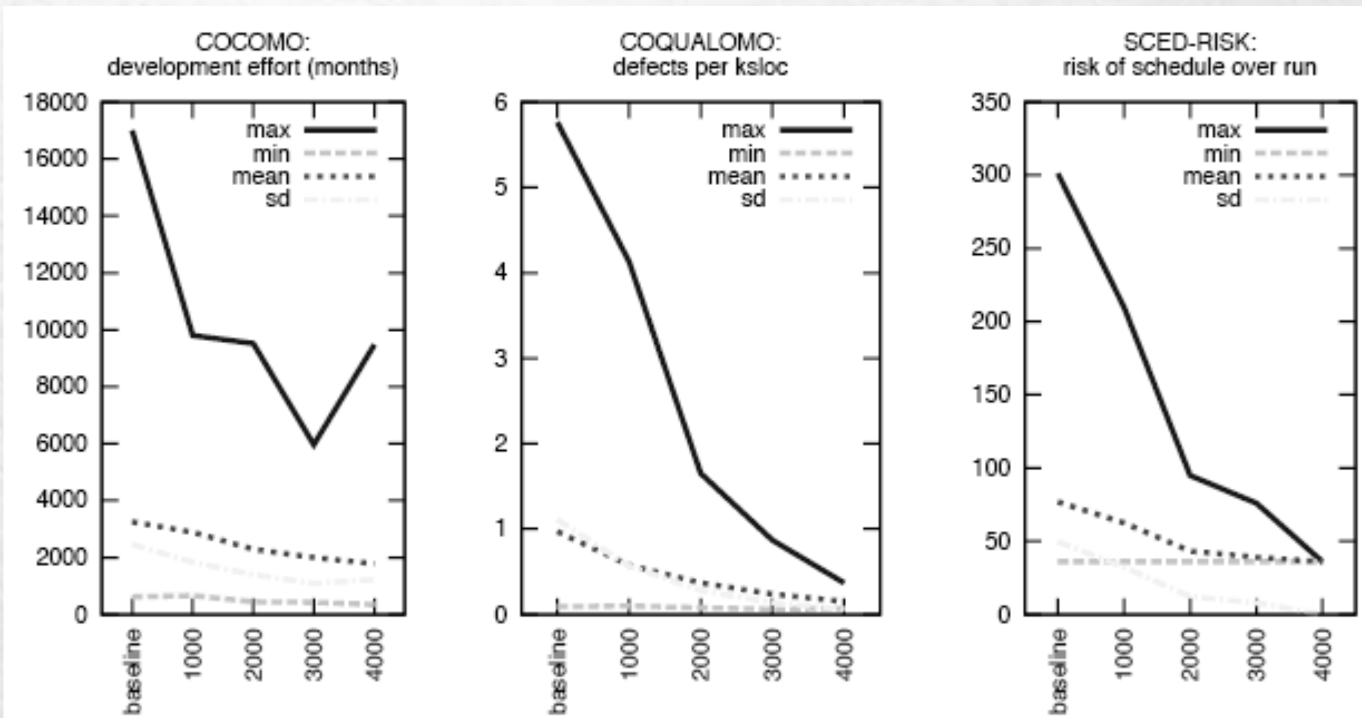
Cycle: repeat till happy (or no more improvement)

Monte carlo simulations

scenario → sample possible inputs → {COCOMO-II, COQUALMO, MADACHY} → BORE: classification → TAR3: learning → (learned restraints) → sample possible inputs



	learned restraints			
baseline	1000	2000	3000	4000
$75 \leq$ ksloc ≤ 125 rely = 5 prec = 1 acap = 5 aexp = 1 cplx = 6 ltex = 1 ruse = 6	sced=4 peer_reviews=5	pmat=5 pcap=4	tool=4 execution_testing- _and_tools=5	team=5 resl=5 automated- analysis=5



variances reduced

Only 17/28 restrained

Sound bites (again)

- 21st software uses autonomy. Are you ready?

- AI software is still software
  - Autonomy= new software
  - But can be analyzed, at least partially, by existing methods

- SE has much to offer AI

- AI has much to offer SE
  - Use data miners to explore COCOMO-model(s)
  - Large scale, easy, what-if scenarios

- If you can't mine data, grow it.

- Monte Carlo + data miners = good
  - Don't just auto-generate
  - Also auto-understand
  - seek the "diamonds in the dust"

- Conclusions without local calibration
  - Seek stable conclusions within the envelope of options

Questions?

Comments?