CIG 2011 Tutorial:
August 31, 2011

Experimentation in CI-Affected Games Research

Mike Preuss

Technische Universität Dortmund
Computational Intelligence Group, Computer Science 11
CI in Games: What is this good for?

With respect to experimentation

last year’s CIG (Alex Champandard):

- some researchers interested in industry problems, many not interested
- industry not too interested in us
- I would argue: normal situation, even worse in other academic fields
- Alex working on this at the Paris Game/AI conference
- we are not the slaves of industry (at least not all of us)
- which enables us to do some fundamental research
We believe
what do we believe?

but of course, our research shall *somehow* make sense:

- we are here because we believe it does
- we are not alone, e.g. AIIDE: different methods/approaches, but similar focus

back to Lucas/Kendall 2006 ”Evolutionary Computation and Games” (IEEE Computational Intelligence Magazine):

1. good testbed to apply our methods
2. do things in a better way
3. do things we (or they) could not do before
the testbed argument seems to loose importance:

- test problem collections (benchmarks) and competitions are getting popular in many fields
- not really simple to transfer back obtained knowledge (games research partly engineering)
- the need to *defend* games research is shrinking
Improvement

the doing things better argument is (still) important:

- can involve theory, but usually based on experimentation
- question: what does better mean?
- measurement sometimes fully automated, sometimes requires user interaction (no fun formula)
- required: being open to other methods (to achieve meaningful comparisons)
- ideal situation: competition as joint effort experiment (fair)
To boldly go... we may encounter problems not solved or not even realized by others:

- interesting features of CI techniques: coping with noise, black-box approach, realtime ability, multiple objectives
- show that our approach indeed does fulfill some minimal requirements by experiment
Algorithm development and theory

- of course we can improve our *methods* while applying them
- but this is usually not restricted to games problems
- improvement/improved understanding may result in better theory
- discrete state games: algorithm engineering cycle applicable
- more complex games (e.g. RTS): theory connection very difficult
- solving games problems is to a large extent *engineering*
- *we have to rely on good experimentation in most cases*
What is an experiment?

we ask Wikipedia (not that this is always right):

An experiment is a method of testing - with the goal of explaining - the nature of reality. [...] More formally, an experiment is a methodical procedure carried out with the goal of verifying, falsifying, or establishing the accuracy of a hypothesis.

important words: goal, reality, methodical procedure, hypothesis
Why do we need experimentation?

- practitioners need to solve problems, even if theory is not developed far enough
- counterargument of practitioners: Tried that once, didn’t work (expertise needed to apply convincingly)
- we need to establish guidelines how to adapt the algorithms to practical problems
- helps theoreticians to find exploitable (problem/method) relations

Experimental methodology is improving, leaving the phase of

a) funny but useless performance figures
b) lots of better and better algorithms that soon disappear again
Why do we need experimentation?

- practitioners need so solve problems, even if theory is not developed far enough
- counterargument of practitioners: Tried that once, didn’t work (expertise needed to apply convincingly)
- we need to establish guidelines how to adapt the algorithms to practical problems
- helps theoreticians to find exploitable (problem/method) relations

instead, we converge to

a) deliberate and justified choice of parameters, problems, performance criteria—much less arbitrariness
b) better generalizability (not quite resolved, but targetted)
Are we alone (with this problem)?

In natural sciences, experimentation is not in question

- Many inventions (batteries, x-rays) made by experimentation, often unintentional
- Experimentation leads to theory, theory has to be useful (can we do predictions?)

Different situation in computer science

- 2 widespread stereotypes influence our view of computer experiments:
  a) Programs do (exactly) what algorithms specify
  b) Computers (programs) are deterministic, so why statistics?
Lessons from other sciences

in economics, experimentation was established quite recently (compared to its age)

- modeling human behavior as rationality assumption (of former theories) had failed
- no accepted new model available: experimentation came in as substitute

in (evolutionary) biology, experimentation and theory building both have problems

- active experimentation only possible in special cases, otherwise only observation
- mainly concepts (working principles) instead of theories: always exceptions
  ⇒ stochastical distributions, population thinking

nonlinear behavior

Ernst Mayr
Experimentation at unexpected places

since \(\approx\)1960s: Experimental Archaeology

- gather (e.g. performance) data that is not available otherwise
- task: concept validation, fill conceptual holes

experimentation in management of technology and product innovation

- product cycles are sped up by ‘fail-fast’, ‘fail-often’ experimentation
- what-if questions may be asked by using improved computational resources
- innovation processes have to be tailored towards experimentation
Algorithm Engineering
How theoreticians handle it...(recently)

- Algorithm Engineering is *theory* + real data + concrete implementations + experiments
- principal reason for experiments: test validity of theoretical claims
- are there important factors in practice that did not go into theory?
- approach also makes sense for CI methods, but we start with no or little theory
- performance measuring usually no problem for us, but user interaction
So what about statistics?

are the methods all there? some are, but:

- our data is usually not normal
- we can most often have lots of data
- this holds for algorithmics, also!
- these are not the conditions statisticians are used to
- in some situations, there is just no suitable test procedure

⇒ there is a need for more statistics and more statistical methods.

Catherine McGeogh:

*our problems are unfortunately not sexy enough for the statisticians...*
What experiments are all about

good experiments all have in common:

- fairness (even if we want to show that our method is better)
- openness, enable the system to come up with surprises
- defined goals
  - how is the winning method identified (comparison)
  - how is reaching minimal requirements defined (study)
- defined procedure (not ad-hoc during experiment)
- documentation (enables others to rebuild experiment)
- iteration (the first question is almost never very good)
Openness example: clutch control in TORCS

EvoStar 2011: Mr Racer bot (Jan Quadflieg, Tim Delbrügger, Mike Preuss) shows potential but has bad clutch control

- first approach similar to Autopia clutch control: speed based
- Autopia closes clutch at below 70 km/h
- we adapt closing (generalized logistic) function with a bit more freedom
- result: using the clutch until 180 km/h is profitable!
- we would be much worse with restriction to 70 km/h

(see the videos)
The hypothesis issue

we remember: *hypothesis* and *goals* important words in experiment definition

Cohens investigation of 1990 (all papers of the AAAI conference):

- almost no relationship between experiments and theory
- 60% testing only on one problem instance
- 80% did not explain the measured result in any way
- 16% provided a hypothesis or defined objectives

Paul R. Cohen
Research question

- not trivial $\Rightarrow$ many papers are not focused
- the (real) question is not: is my algorithm faster than others on a set of benchmark problems?
- final task is *reality*, not benchmarking

*horse racing*: set up, run, comment...

explaining observations leads to new questions:

- explanations we give for a result can often also be tested experimentally
- range of validity shall be explored (problems, timing, parameters, etc.)
Research question

- not trivial ⇒ many papers are not focused
- the (real) question is not: is my algorithm faster than others on a set of benchmark problems?
- final task is reality, not benchmarking

  *horse racing*: set up, run, comment... NO!

explaining observations leads to new questions:

- explanations we give for a result can often also be tested experimentally
- range of validity shall be explored (problems, timing, parameters, etc.)
How to set up research questions?

- when comparing, always ask if a difference is meaningful in practice
- same for experimental studies: what quality level is required to make the approach useful?
- usually, we do not know the ‘perfect question’ from the start, this requires some experimentation...
- an inherent problem with experimentation is that we do (should) not know the outcome in advance
- but it may lead to new, better questions
- try small steps, expect the unexpected
how do we obtain decision criteria from an initial research question?

- set up scientific claims
- formulate as statistical hypotheses
- experiment, and then the same way back
Components of an (optimization type) experiment

- Algorithm (program)
- Parameter set
- Test problem
- Performance measure
- Termination criterion
- Initialization

Induces:
- Control flow
- Data flow

Problem design:
- Test problem
- Performance measure
- Termination criterion
- Initialization
Factors: overview

- Java version
- Hardware
- Color of experimenter's socks
- Weather

EXPECTED
- Algorithm design
- Problem design
- Operating system

POSSIBLE, UNWANTED
- Color of experimenter's socks
- Weather

UNEXPECTED
- Room temperature
Tuning: when there are too many factors

- comparison of methods without suitable parameter settings is comparing unsuitable algorithms
- not looking at parameters often means to give away good performance
- tuning reveals parameter relevance and interactions

even if time is critical: small design with 10 points in parameter space already reveals a lot
Tuning: available methods

recent compilation of methods and more general considerations concerning experimental approaches

methods:

- SPO (Bartz-Beielstein): sequential model-based improvement
- F-Race (Birattari, Stützle): iterative bad parameter elimination
- REVAC (Nannen, Eiben, Smit): meta-EDA
- ParamILS (Stützle, Hoos, Hutter): iterative local search
- probably more to come, active research area...
Reporting and keeping track of experiments

around 40 years of experimental tradition in CI, but:

- no standard scheme for reporting experiments (experimental protocols)
- instead: one (“Experiments”) or two (“Experimental Setup” and “Results”) sections in papers, often providing a bunch of largely unordered information
- affects readability and impairs reproducibility

keeping experimental journals helps:

- record context and rough idea
- report each experiment
- running where (machine)
- finished when (date/time), link to result file(s)

⇒ we suggest a 7-part reporting scheme
Suggested report structure

ER-1: **research question** the matter dealt with

ER-2: **pre-experimental planning** first—possibly explorative—program runs, leading to task and setup

ER-3: **task** main question and scientific and derived statistical hypotheses to test

ER-4: **setup** problem and algorithm designs, sufficient to replicate an experiment

ER-5: **results/visualization** raw or produced (filtered) data and basic visualizations

ER-6: **observations** exceptions from the expected, or unusual patterns noticed, plus additional visualizations, no subjective assessment

ER-7: **discussion** stat-test results and necessarily subjective interpretations for data and especially observations
Statistical testing

- many papers now employ statistical testing
- but we claim: fundamental ideas from statistics are misunderstood!
- for example: what is the $p$ value?

Definition ($p$ value)

the $p$ value is the probability that the null hypothesis is true
Statistical testing

- many papers now employ statistical testing
- but we claim: fundamental ideas from statistics are misunderstood!
- for example: what is the $p$ value?

**Definition ($p$ value)**

the $p$ value is the probability that the null hypothesis is true. **No!**
many papers now employ statistical testing

but we claim: fundamental ideas from statistics are misunderstood!

for example: what is the $p$ value?

**Definition ($p$ value)**

the $p$ value is

$$p = P\{ \text{obtain observed result, or greater} \mid \text{null model is true} \}$$

⇒ the $p$ value is not related to any probability whether the null hypothesis is true or false
To test or not to test?

yes, but:

- we often have non-normal data
  ⇒ non-parametric tests, permutation tests

- temptation to “make” tests valid by enlarging sample (not always helpful, e.g. if distribution bimodal)
  ⇒ rule-of-thumb fixed size (e.g. 30)
Wilcoxon rank sum test

- also Mann-Whitney U-test or just U-test (equivalent)
- more robust than t-test, becoming standard test in Evolutionary Computation
- basic assumption: distribution functions $G$ and $F$ of $X$ and $Y$ only differ by a shift $a$, $G(x) = F(x - a)$
- this also means homogeneity of variances (may require F-test)!
- null hypothesis: $H_0 : a = 0$, $H_1 : a \neq 0$
- R-command:
  
  ```
  wilcox.test(x, y, alternative = "two.sided",
  conf.level = 0.95)
  ```
A simple example

(rexp() gives random numbers from an exponential distribution)

> N=10
> X=rexp(N)
> X
[1] 0.51762849 1.20825633 3.23399265 1.80257160 0.85732474
0.24931676 0.48776898 0.81129961 0.70829536 0.02036845
> Y=rexp(N)+0.2
> Y
[1] 0.457792 2.224912 1.095469 1.224541 5.392600 2.577539
2.334396 1.235689 7.564990 1.420925
> wilcox.test(X,Y)
Wilcoxon rank sum test
data: X and Y
W = 20, p-value = 0.02323
alternative hypothesis: true location shift is not equal to 0
Simple example part 2

the same with a t-test:

```r
> t.test(X,Y)
```

Welch Two Sample t-test

data: X and Y
t = -2.0473, df = 12.066, p-value = 0.06305
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
-3.22584888 0.09944275
sample estimates:
mean of x mean of y
0.9896823 2.5528854

But: much less difference in test results if distributions are a bit more normal
Earth is round \( (p < 0.05) \)

- paper of Jacob Cohen (in American Psychologist, 1994)
- summarizes criticism on 'unreflected' use of statistical testing
- be careful with small samples!
- first understand and improve data (EDA, Exploratory Data Analysis, after Tukey), then testing
- actually, one should test the other way around: postulate null hypothesis and try to falsify it (very time-consuming procedure)
- providing confidence intervals gives important information!
- importance of reproducing a result
Correlations and correlation tests

- correlation coefficient: measure for (linear) relation of two measurements of same sample, between $+1$ (ideally correlated) and $-1$ (anti-correlated)

- example (work with Jan Quadflieg and Günter Rudolph): TORCS, 2-round times for 2 tracks in sec., different 'AI-drivers'

- question: are good drivers on track 1 also good on track 2?

```
in R (data in nd):
cor(nd$track1,nd$track2)
> [1] -0.075339
```

looks uncorrelated. really?

<table>
<thead>
<tr>
<th>ID</th>
<th>track1</th>
<th>track2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1137</td>
<td>441.4660</td>
<td>362.246</td>
</tr>
<tr>
<td>1069</td>
<td>438.6060</td>
<td>363.466</td>
</tr>
<tr>
<td>1059</td>
<td>437.8260</td>
<td>361.886</td>
</tr>
<tr>
<td>1027</td>
<td>438.3060</td>
<td>362.166</td>
</tr>
<tr>
<td>1162</td>
<td>439.7460</td>
<td>361.746</td>
</tr>
</tbody>
</table>

etc, 75 drivers
Different correlations

- default method is Pearson correlation (uses actual values)
- but: we assume *linear* relation of the measurements!
- alternatives: rank correlations (non-parametric)
  - Spearman’s rank correlation coefficient
    (as Pearson correlation, but using ranks)
  - Kendall’s tau ($\tau$): uses no rank differences but only relative positions (less sensitive to outliers)
Correlated or not?

applying non-parametric correlation measures results in:

```r
> cor(nd$track1,nd$track2,method="spearman")
> [1] -0.6390533
> cor(nd$track1,nd$track2,method="kendall")
> [1] -0.4619204
```

looks like a strong correlation
(rule of thumb from psychology: $|\text{cor}| > 0.4$ is strong)

good drivers of track 1 usually perform worse on track 2 and vice versa
Correlation test

same data as before, just `cor.test` instead of `cor`:

```r
> cor.test(nd$track1, nd$track2, method = "kendall")
Kendall's rank correlation tau
data: nd$track1 and nd$track2
z = -5.6756, p-value = 1.382e-08
alternative hypothesis: true tau is not equal to 0
sample estimates:
tau
-0.4619204
```
Floor and ceiling effects

- **floor effect**: compared methods attain set task very rarely
  ⇒ problem is too hard

- **ceiling effect**: methods nearly always reach given task
  ⇒ problem is too easy

If problem is too hard or too easy, nothing is shown.

- Pre-experimentation is necessary to obtain reasonable tasks

- If task is reasonable (e.g. practical requirements), then algorithms are unsuitable (floor) or all good enough (ceiling), statistical testing does not provide more information

- Arguing on minimal differences is statistically unsupported and scientifically meaningless
Confounded effects

two or more effects or helper algorithms are merged into a new technique, which is improved

- where does the improvement come from?
- it is necessary to test both single effects/algorithms, too
- either the combination helps, or only one of them
- knowing that is useful for other researchers!
Underestimated randomness

- idea: find pareto front of two tuning criteria
- parameter changes not interpretable
- validation failed
- reason: deviations much too high!

more difficulties: see also papers of the GECCO’09 workshop

*Learning from Failures in Evolutionary Computation (LFFEC)*
There is a problem with the experiment

after all data is in, we realize that something was wrong (code, parameters, environment?), what to do?

- current approach: either do not mention it, or redo everything
- if redoing is easy, nothing is lost
- if it is not, we must either:
  - let people know about it, explaining why it probably does not change results
  - or do validation on a smaller subset: how large is the difference (e.g. statistically significant)?
- do not worry, this situation is rather normal
- *Thomke*: there is nearly always a problem with an experiment
- early experimentation reduces the danger of something going completely wrong
What are the objectives?

difficult to say, at some point user is involved
(fun: Georgios’ and Julian’s problem)

some approaches:

- completely interactive (user takes all decisions), usually preference based
- model-based, model is learned from user data and used for decisions
- some mix of the two (partially interactive)
- interesting: EvoMusart people have same problem (automated assessment of aesthetic criteria)
Or bottom up: make up many objectives

example: multiobjective StarCraft map-making
(Togelius, Preuss, Beume, Wessing, Hagelbäck, Yannakakis)

- 8 objectives related to base location, ressource fairness, path properties (e.g. choke points)
- unclear which objectives make sense
- but single objectives can be discussed with users
- users may be wrong (e.g. one can make fair and asymmetric maps)
- tool that helps exploring: multi-objective optimization
If fun is individual, so is believability

(from joint work with Markus Kemmerling and Niels Ackermann)

- indirect learning of player believability preferences in Diplomacy bots
- several test games per player, submitted believability ranking of 6 other players (bots)
- find best matching between rankings and measured features (minimizing rank errors by adjusting weights)

![Graph showing module weights for believability preferences.](image)
Diagrams instead of tables
Visualizing high-dimensional data: MDS

- for more than 3D, we need a data reduction
- but we want to recognize the inner relation in the data
- one possibility: multi-dimensional scaling (MDS)
- uses only a distance matrix of the data
- more or less equivalent to principal component analysis (PCA)
- applying clustering to the results usually makes sense

(following example from joint work with Phil Hingston)
MDS example: Red Teaming strategies

- **Red Teaming**: detect attack (red) strategies that prevail against an existing defense (blue) strategy
- **RedTNet**: simple (in this case grid) node structure, 2 teams (red, blue) with several agents
- Moves simultaneous (to neighbor node), majority in a node kills minority
- Green nodes need to be conquered to win game
- Strategies coded as node sequences, optimized via co-evolution
MDS visualization in R

- distance matrices generated for red and blue final populations (in this case similar to edit distance)
- produce MDS in R (for 2D):
  ```r
  fit <- cmdscale((distmatrix), k=2)
  ```
- plotting: `plot(fit$points[,1], fit$points[,2])`

![MDS visualization](image)

**blue, popsize=20, testsize=10, crowding=false**

**red, popsize=20, testsize=10, crowding=false**
Modeling away misleading sensor information

(joint work with Jan Quadflieg and Simon Wessing)

a) to d): approaching a hairpin corner (left), and a full speed corner (right), with sensory input

Jan developed a geometric method to extract a curvature value but in 2010, competition organisers added noise (10%) to the sensor values, which broke our method
Kriging model of sensory data

- 19 (noisy) 'forward' sensors between 0 and 200m
- data sampled every 20ms, some accumulation (up to 10 values) tolerable
  \( \Rightarrow \) Noise drops to \( \approx 3\% \)
- longer delay not tolerable, we need to act!

Preliminary results:
- model input randomly drawn from one round \( \approx 20,000 \) data
- less than 400 input points not sufficient for model that copes with 'some' noise
- we use the \textit{DiceKriging} (R) package
- modeling the non-noisy data works well
And now with noise...

- validation with 1000 randomly selected of the 20,000 points
- works incredibly well for a naive approach (no expert knowledge)
- Kriging interesting tool for obtaining a good model quickly
- further improvement simple: pre-selection of learning data
Car setup optimization problem modeled

22 normalized real values for tuning car:

- gear ratios
- angles of front/rear wing
- brake system
- anti-roll bars
- wheels
- suspension (springs, ride height)

competition setup allows for 500 samples (with 2000 tics) max
Some methods compared

(joint work with David Ginsbourger and Tobias Wagner)

we distribute 500 points in 22D via Latin Hypercube Design (LHD), results sorted according to root mean squared error (RMSE), measured on (accurate) validation set:

<table>
<thead>
<tr>
<th>method</th>
<th>repeats</th>
<th>RMSE</th>
<th>RMSEstd</th>
</tr>
</thead>
<tbody>
<tr>
<td>kriging-gauss</td>
<td>10</td>
<td>0.0936</td>
<td>0.0000</td>
</tr>
<tr>
<td>gam</td>
<td>1</td>
<td>0.0976</td>
<td>0.0000</td>
</tr>
<tr>
<td>randomForest</td>
<td>10</td>
<td>0.1069</td>
<td>0.0008</td>
</tr>
<tr>
<td>rpart</td>
<td>1</td>
<td>0.1140</td>
<td>0.0000</td>
</tr>
<tr>
<td>svm-radial</td>
<td>1</td>
<td>0.1515</td>
<td>0.0000</td>
</tr>
<tr>
<td>lm</td>
<td>1</td>
<td>0.1699</td>
<td>0.0000</td>
</tr>
<tr>
<td>svm-linear</td>
<td>1</td>
<td>0.1718</td>
<td>0.0000</td>
</tr>
</tbody>
</table>

most interesting methods: kriging, gam, randomForest, rpart ok
Summary

- being aware of utilized experimental methodology is important
- more exploration possible (openness): let the system decide
- experiments are not perfect, iterate and improve
- statistics helpful, better do non-parametric
- modeling works even for 20D, but special tools needed for preference relation data
- visualization incredibly important