Machine Learning, Probabilistic Inference, System Identification and Control

Sheffield Gaussian Process Summer School, 2014

Carl Edward Rasmussen

Department of Engineering, University of Cambridge

September 15-17th, 2014

with thanks to my students:
Marc Deisenroth, Roger Frigola, Joe Hall & Andrew McHutchon

Adaptive Systems Wish List

Good adaptive systems must be

- flexible
- automatic
- efficient
- able to handle noise and uncertainty
- practical
- optimal
- provably stable

Adaptive Systems Wish List – Realistic Version

Good adaptive systems must be

- flexible
- automatic
- efficient
- able to handle noise and uncertainty
- practical
- optimal
- provably stable

Probabilistic or Bayesian Inference

Probability theory is a framework ideally suited to make inferences under uncertainty

- able to handle noise and uncertainty
- automatic
- efficient,

but, can it be made

- flexible?
- practical?

Unfortunately, the Bayesian framework is often misunderstood.

Be flexible: Don't Lie

Cromwell's dictum:

> *I beseech you, in the bowels of Christ, think it possible that you may be mistaken*

In modeling terms

- don't assign zero probability to something that *might* be true
  Ex.: 'The friction is assumed purely viscous'.
- don't condition on things which *might not* be true.
  Ex.: 'The Kalman filter is optimal for linear Gaussian systems'.

How can we achieve this? Non-parametric inference over functions

Learning Dynamics: Short and Long time Horizons

It is only tractable to capture *short term* dynamics …

… but you need to understand *long term* dynamics to control

$\Rightarrow$

- learn short term dynamics
- make probabilistic predictions (because we don't know the *exact* dynamics)
- probabilistically cascade predictions to get long term behaviour
- propagating a Gaussian state distribution through a non-linear dynamics model is intractable: use moment matching

Learning Algorithm

Require:

- reward or loss function
- initial policy or control law (random)

Algorithm:

- execute current algorithm in the real world, collect data
- train the short term dynamics models on all available data
- evaluate different controllers by *simulating* their long term performance
- pick the best controller you can find (using gradient based minimization)
- repeat

Cart and Pole

Loss function geometric (not dynamic).

Short term (100ms) dynmics are captured by 4 seperate GP models.

A non-linear parameterized state feedback policy is optimized based on the experience so far.

The system learns quickly and automatically, from essentially
no prior knowledge:

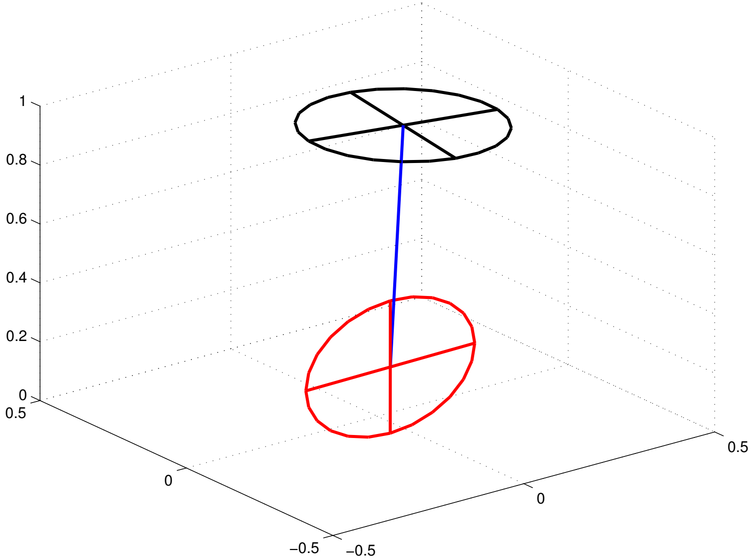
- relevant time frames: sampling time 100 ms, horizon 2 s.
- dynamics are smooth
- dynamics are time-invariant
- scale for the loss function

Properties of the Solution

Some observations about the learner

- Dynamics are learnt only locally around successful trajectories
- learning algorithm is greedy
  - exploration vs exploitation
- characteristic behaviour of the uncertainty or error-bars
  - Initially, error bars are wide and expanding
  - after successful learning, error bars are collapsing

Learning to Unicycle



Equations of Motion

```
A = [                                                                                                                                -Ct*sf
                                                                                                                                             Ct*cf*ct
                                                                                            0                                                                0    Ct;
                                                                                                                                  0
                                                                      Cw*st+At*st-rf*(-mf*(st*rf+cf*st*rw)-mt*(st*r+cf*st*rw))+rt*mt*(st*r+cf*st*rw)
                                                                     -cf*rw*(rf*(mf+mt)+rt*mt)                     -Cw-At-rf*(mf*rf+mt*r)-rt*mt*r      0;
                                                 cf*(-Af*sf-Ct*sf)-sf*(-Bf*cf-At*cf+rf*(-mf*(cf*rf+rw)-mt*(cf*r+rw))-rt*mt*(cf*r+rw))
                                                     Aw*ct+cf*(Af*cf*ct+Ct*cf*ct)-sf*(-Bf*sf*ct-At*sf*ct+rf*(-mf*sf*ct*rf-mt*sf*ct*r)-rt*mt*sf*ct*r)
                                                                                            0                                                                0 Ct*cf;
  -Aw-rw*(mf*(cf*rf+rw)+mw*rw+mt*(cf*r+rw))+sf*(-Af*sf-Ct*sf)+cf*(-Bf*cf-At*cf+rf*(-mf*(cf*rf+rw)-mt*(cf*r+rw))-rt*mt*(cf*r+rw))
                             -rw*(mt*sf*ct*r+mf*sf*ct*rf)+sf*(Af*cf*ct+Ct*cf*ct)+cf*(-Bf*sf*ct-At*sf*ct+rf*(-mf*sf*ct*rf-mt*sf*ct*r)-rt*mt*sf*ct*r)
                                                                                            0                                                                0 Ct*sf;
                                                                                                                                  0 2*Cw*st+At*st-rf*(-mt*(s
t*r+cf*st*rw)-mf*(st*rf+cf*st*rw))+rt*mt*(st*r+cf*st*rw)+rw*(mw*st*rw+sf*(mf*sf*st*rw+mt*sf*st*rw)+cf*(mt*(st*r+cf*st*rw)+mf*(st*rf+cf*st*rw))) -Cw-rt*mt
*cf*rw+rw*(-mw*rw+sf*(-mf*sf*rw-mt*sf*rw)+cf*(-mf*cf*rw-mt*cf*rw))-rf*(mt*cf*rw+mf*cf*rw) -Cw-At-rf*(mf*rf+mt*r)-rt*mt*r-rw*cf*(mf*rf+mt*r)       0 ];

b = zeros(5,1);
b(1) = -V(t)+Ct*(-dphi*sf*dpsif*ct-dphi*cf*st*dtheta-cf*dpsif*dtheta);
b(2) = -U(t)+Cw*dphi*ct*dtheta-(-dphi*cf*ct+sf*dtheta)*Bf*(dphi*sf*ct+cf*dtheta)+(dphi*sf*ct+cf*dtheta)*Af*(-dphi*cf*ct+sf*dtheta)+At*dphi*ct*dtheta-(dph
i*sf*ct+cf*dtheta)*Ct*(dphi*cf*ct-sf*dtheta+dpsit)+(dphi*cf*ct-sf*dtheta)*At*(dphi*sf*ct+cf*dtheta)-rf*(-mf*g*sf*ct-mf*(sf*dpsif*(-dphi*st+dpsiw)*rw+cf*d
phi*ct*dtheta*rw-(-dphi*cf*ct+sf*dtheta)*(dtheta*rw+(dphi*sf*ct+cf*dtheta)*rf)+dphi*ct*dtheta*rf-(-dphi*st+dpsif)*sf*(-dphi*st+dpsiw)*rw)-mt*g*sf*ct-mt*(
sf*dpsif*(-dphi*st+dpsiw)*rw+cf*dphi*ct*dtheta*rw-(-dphi*st+dpsif)*sf*(-dphi*st+dpsiw)*rw+dphi*ct*dtheta*(rf+rt)+(dphi*cf*ct-sf*dtheta)*(dtheta*rw+(dphi*
sf*ct+cf*dtheta)*(rf+rt))))-rt*(-mt*g*sf*ct-mt*(sf*dpsif*(-dphi*st+dpsiw)*rw+cf*dphi*ct*dtheta*rw-(-dphi*st+dpsif)*sf*(-dphi*st+dpsiw)*rw+dphi*ct*dtheta*
(rf+rt)+(dphi*cf*ct-sf*dtheta)*(dtheta*rw+(dphi*sf*ct+cf*dtheta)*(rf+rt))));
b(3) = -T*ct-2*dphi*st*Aw*dtheta-dtheta*Cw*(-dphi*st+dpsiw)+cf*(-Af*(dphi*sf*dpsif*ct+dphi*cf*st*dtheta+cf*dpsif*dtheta)-(dphi*sf*ct+cf*dtheta)*Cf*(-dphi
*st+dpsif)+(-dphi*st+dpsif)*Bf*(dphi*sf*ct+cf*dtheta)+Ct*(-dphi*sf*dpsif*ct-dphi*cf*st*dtheta-cf*dpsif*dtheta))-sf*(-Bf*(dphi*cf*dpsif*ct-dphi*sf*st*dthe
ta-dpsif*sf*dtheta)-(-dphi*st+dpsif)*Af*(-dphi*cf*ct+sf*dtheta)+(-dphi*cf*ct+sf*dtheta)*Cf*(-dphi*st+dpsif)-At*(dphi*cf*dpsif*ct-dphi*sf*st*dtheta-dpsif*
sf*dtheta)-(dphi*cf*ct-sf*dtheta)*At*(-dphi*st+dpsif)+(-dphi*st+dpsif)*Ct*(dphi*cf*ct-sf*dtheta+dpsit)+rf*(mf*g*st-mf*((dphi*sf*ct+cf*dtheta)*sf*(-dphi*s
t+dpsiw)*rw+(dphi*cf*dpsif*ct-dphi*sf*st*dtheta-dpsif*sf*dtheta)*rf+(-dphi*cf*ct+sf*dtheta)*(-cf*(-dphi*st+dpsiw)*rw-(-dphi*st+dpsif)*rf))+mt*g*st-mt*(-(
dphi*cf*ct-sf*dtheta)*(-cf*(-dphi*st+dpsiw)*rw-(-dphi*st+dpsif)*(rf+rt))+(dphi*cf*dpsif*ct-dphi*sf*st*dtheta-dpsif*sf*dtheta)*(rf+rt)+(dphi*sf*ct+cf*dthe
ta)*sf*(-dphi*st+dpsiw)*rw))+rt*(mt*g*st-mt*(-(dphi*cf*ct-sf*dtheta)*(-cf*(-dphi*st+dpsiw)*rw-(-dphi*st+dpsif)*(rf+rt))+(dphi*cf*dpsif*ct-dphi*sf*st*dthe
ta-dpsif*sf*dtheta)*(rf+rt)+(dphi*sf*ct+cf*dtheta)*sf*(-dphi*st+dpsiw)*rw)));
b(4) = -dphi^2*st*Aw*ct-dphi*ct*Cw*(-dphi*st+dpsiw)-rw*(mw*dphi*ct*(-dphi*st+dpsiw)*rw-mt*g*st-mw*g*st+mf*((dphi*sf*ct+cf*dtheta)*sf*(-dphi*st+dpsiw)*rw+
(dphi*cf*dpsif*ct-dphi*sf*st*dtheta-dpsif*sf*dtheta)*rf+(-dphi*cf*ct+sf*dtheta)*(-cf*(-dphi*st+dpsiw)*rw-(-dphi*st+dpsif)*rf))-mf*g*st+mt*(-(dphi*cf*ct-s
f*dtheta)*(-cf*(-dphi*st+dpsiw)*rw-(-dphi*st+dpsif)*(rf+rt))+(dphi*cf*dpsif*ct-dphi*sf*st*dtheta-dpsif*sf*dtheta)*(rf+rt)+(dphi*sf*ct+cf*dtheta)*sf*(-dph
i*st+dpsiw)*rw))+sf*(-Af*(dphi*sf*dpsif*ct+dphi*cf*st*dtheta+cf*dpsif*dtheta)-(dphi*sf*ct+cf*dtheta)*Cf*(-dphi*st+dpsif)+(-dphi*st+dpsif)*Bf*(dphi*sf*ct+
cf*dtheta)+Ct*(-dphi*sf*dpsif*ct-dphi*cf*st*dtheta-cf*dpsif*dtheta))+cf*(-Bf*(dphi*cf*dpsif*ct-dphi*sf*st*dtheta-dpsif*sf*dtheta)-(-dphi*st+dpsif)*Af*(-d
phi*cf*ct+sf*dtheta)+(-dphi*cf*ct+sf*dtheta)*Cf*(-dphi*st+dpsif)-At*(dphi*cf*dpsif*ct-dphi*sf*st*dtheta-dpsif*sf*dtheta)-(dphi*cf*ct-sf*dtheta)*At*(-dphi
*st+dpsif)+(-dphi*st+dpsif)*Ct*(dphi*cf*ct-sf*dtheta+dpsit)+rf*(mf*g*st-mf*((dphi*sf*ct+cf*dtheta)*sf*(-dphi*st+dpsiw)*rw+(dphi*cf*dpsif*ct-dphi*sf*st*dt
heta-dpsif*sf*dtheta)*rf+(-dphi*cf*ct+sf*dtheta)*(-cf*(-dphi*st+dpsiw)*rw-(-dphi*st+dpsif)*rf))+mt*g*st-mt*(-(dphi*cf*ct-sf*dtheta)*(-cf*(-dphi*st+dpsiw)
*rw-(-dphi*st+dpsif)*(rf+rt))+(dphi*cf*dpsif*ct-dphi*sf*st*dtheta-dpsif*sf*dtheta)*(rf+rt)+(dphi*sf*ct+cf*dtheta)*sf*(-dphi*st+dpsiw)*rw))+rt*(mt*g*st-mt
*(-(dphi*cf*ct-sf*dtheta)*(-cf*(-dphi*st+dpsiw)*rw-(-dphi*st+dpsif)*(rf+rt))+(dphi*cf*dpsif*ct-dphi*sf*st*dtheta-dpsif*sf*dtheta)*(rf+rt)+(dphi*sf*ct+cf*
dtheta)*sf*(-dphi*st+dpsiw)*rw)));
b(5) = -T*st+2*Cw*dphi*ct*dtheta+(dphi*sf*ct+cf*dtheta)*Af*(-dphi*cf*ct+sf*dtheta)-rt*(-mt*g*sf*ct-mt*(sf*dpsif*(-dphi*st+dpsiw)*rw+cf*dphi*ct*dtheta*rw-
(-dphi*st+dpsif)*sf*(-dphi*st+dpsiw)*rw+dphi*ct*dtheta*(rf+rt)+(dphi*cf*ct-sf*dtheta)*(dtheta*rw+(dphi*sf*ct+cf*dtheta)*(rf+rt))))-(dphi*sf*ct+cf*dtheta)
*Ct*(dphi*cf*ct-sf*dtheta+dpsit)+At*dphi*ct*dtheta+rw*(2*mw*rw*dphi*ct*dtheta+sf*(mf*(-cf*dpsif*(-dphi*st+dpsiw)*rw+sf*dphi*ct*dtheta*rw+(dphi*sf*ct+cf*d
theta)*(dtheta*rw+(dphi*sf*ct+cf*dtheta)*rf)-(-dphi*st+dpsif)*(-cf*(-dphi*st+dpsiw)*rw-(-dphi*st+dpsif)*rf))-mf*g*cf*ct-mt*g*cf*ct-mt*(cf*dpsif*(-dphi*st
+dpsiw)*rw-sf*dphi*ct*dtheta*rw+(-dphi*st+dpsif)*(-cf*(-dphi*st+dpsiw)*rw-(-dphi*st+dpsif)*(rf+rt))-(dphi*sf*ct+cf*dtheta)*(dtheta*rw+(dphi*sf*ct+cf*dthe
ta)*(rf+rt))))+cf*(mf*(sf*dpsif*(-dphi*st+dpsiw)*rw+cf*dphi*ct*dtheta*rw-(-dphi*cf*ct+sf*dtheta)*(dtheta*rw+(dphi*sf*ct+cf*dtheta)*rf)+dphi*ct*dtheta*rf-
(-dphi*st+dpsif)*sf*(-dphi*st+dpsiw)*rw)+mt*g*sf*ct+mf*g*sf*ct+mt*(sf*dpsif*(-dphi*st+dpsiw)*rw+cf*dphi*ct*dtheta*rw-(-dphi*st+dpsif)*sf*(-dphi*st+dpsiw)
*rw+dphi*ct*dtheta*(rf+rt)+(dphi*cf*ct-sf*dtheta)*(dtheta*rw+(dphi*sf*ct+cf*dtheta)*(rf+rt)))))+(dphi*cf*ct-sf*dtheta)*At*(dphi*sf*ct+cf*dtheta)-rf*(-mt*
g*sf*ct-mt*(sf*dpsif*(-dphi*st+dpsiw)*rw+cf*dphi*ct*dtheta*rw-(-dphi*st+dpsif)*sf*(-dphi*st+dpsiw)*rw+dphi*ct*dtheta*(rf+rt)+(dphi*cf*ct-sf*dtheta)*(dthe
ta*rw+(dphi*sf*ct+cf*dtheta)*(rf+rt)))-mf*g*sf*ct-mf*(sf*dpsif*(-dphi*st+dpsiw)*rw+cf*dphi*ct*dtheta*rw-(-dphi*cf*ct+sf*dtheta)*(dtheta*rw+(dphi*sf*ct+cf
*dtheta)*rf)+dphi*ct*dtheta*rf-(-dphi*st+dpsif)*sf*(-dphi*st+dpsiw)*rw))-(-dphi*cf*ct+sf*dtheta)*Bf*(dphi*sf*ct+cf*dtheta);

dz = zeros(14,1);
dz(3:7) = -A\b;
dz(1) = rw*(cos(phi)*dz(5)-sin(phi)*dphi*dpsiw);
dz(2) = rw*(sin(phi)*dz(5)+cos(phi)*dphi*dpsiw);
dz(8:14) = z(1:7);
```

Conclusions

It is possible to learn automatically and rapidly with essentially no prior knowledge.

Don't lie about the complexity of the system: use non-parametric models

Don't lie about uncertainties: faithfully keep track of error-bars.

The learning approach is robust and practical for real applications:

- calibration of sensors and actuators unnecessary
- inaccurate assumptions unnecessary