Exam in Systems of Systems/Complex Systems

Jeppe Nørgaard Jensen and Mathias Clement Frederiksen

CA9 – 2022/2023

Contents

[A1) Problems 1 and 2 3](#_Toc123988857)

[Problem 1 – Prisoner’s dilemma 3](#_Toc123988858)

[Problem 2 – Multipath routing 4](#_Toc123988859)

[A2): Game Theory and Smart Grid 5](#_Toc123988860)

[B1) Reinforcement Learning – stochastic approach 8](#_Toc123988861)

[B2) Model-free Reinforcement Learning – temporal difference, epsilon-greedy exploration, Q-learning 13](#_Toc123988862)

[Temporal difference 13](#_Toc123988863)

[Epsilon Greedy policy 13](#_Toc123988864)

[Q-learning 13](#_Toc123988865)

[C1) Passivity for two systems in negative feedback loop 17](#_Toc123988866)

[C2) Lecture 9: 18](#_Toc123988867)

[Principle of Markov Chain Monte Carlo techniques 19](#_Toc123988868)

[D1) Metropolis Hastings algorithm 19](#_Toc123988869)

[Exercise 1 20](#_Toc123988870)

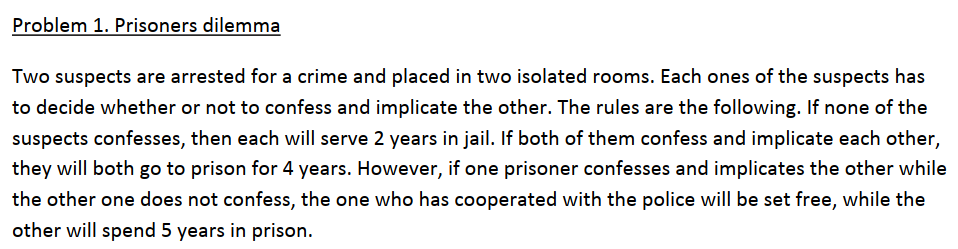
[D2) Gibbs sampling 23](#_Toc123988871)

[D3) Explain how the Conditional Particle Filter with Ancestor Sampling can be used as a Markov Chain Monte Carlo algorithm. 26](#_Toc123988872)

[Exercise 1 26](#_Toc123988873)

# A1) Problems 1 and 2

## Problem 1 – Prisoner’s dilemma



Scenarios:

* If both denies, then each will serve 2 years in jail.
* If both susspect confesses and implicate each other, they will both go to prison for 4 years.
* However, if one prisoner confesses and implicates the other while the other one does not confess, theone who has cooperated with the police will be set free, while the other will spend 5 years in prison.

**Matrix representation of the game  
 Suspect 2**

|  |  |
| --- | --- |
| **Confess** | **Deny** |
| **Confess** | **4 years**  **4 years** | **Free**  **5yers** |
| **Deny** | **5 years**  **Free** | **2 year**  **2years** |

**Suspect 1**

**Define who players in this game are and what the possible strategies are:**

Players are suspect 1 and 2.

**Is this game a zero‐sum game or not‐zero‐sum game?**

Consider (2,2) if suspect 1 confesses, he gets -2, but the other gets +3 hence non-zero-sum game

**Are there any dominating strategies?**

Dominating strategy is the one that gives best payoff: minimizes prison time.

Regardless of the other suspect strategy: Confess minimizes prison length.

**Find Nash equilibrium:**

Deny-Deny is global optimal. But this is an unstable state.

Both suspects confess is a Nash equilibrium!

## Problem 2 – Multipath routing

|  |  |
| --- | --- |
| User 1 would like to send his data packets from B to D   * Can chose routes: B‐A‐D or B‐C‐D   User 2 would like to send his packets from A to C   * Can chose routes: A‐B‐C and A‐D‐C |  |

**Write a matrix to describe this game.**

**User 2**

|  |  |
| --- | --- |
| A‐B‐C | A‐D‐C |
| B‐A‐D | **5+1=6**  **5+3=8** | **3+2=5**  **2+2=4** |
| B-C‐D | **4+2=6**  **3+4=7** | **3+4=7**  **1+4=5** |

**User 1**

**Investigate if the game has Nash equilibrium and what would be a rational choice for the two users.**

Don’t think there is a Nash equilibrium:

User 2 will then always choose A-B-C since this results in more throughout

**User 1** does not care about routes.

# A2): Game Theory and Smart Grid

Players: 2 consumers and

Each may consume either or reps. Of energy each hour

Constraints:

* needs 5 units from between 02 and 10 o’clock
* needs 10 units between 05 and 20 o’clock

Energy prize/unit is calculated hourly and increases affinely with the total demand per hour:

The hourly cost of consumers can be calculated:

The cost function is the daily electricity bill.

**What is the set of strategies for each player?**

* Consume energy, produce energy, or do nothing!

We think that:

will take the required energy when he/she Is alone on the grid between 2 to 5!

* In this period will minimize his own cost by producing energy!

will likewise take the required energy when he/she is alone on the grid between 10 to 20!

* In this period will minimize his own cost by production energy!

## Aggregated method – collaboration (god mode)

Define an aggregated cost. Sum of individual costs:

Constraints and are formulated as:

**Lagrange constrained optimization:**

Gradient of cost function

and dependencies of and :

For time

For time

We can now write up 24 equations of the form

+ 2 constraint equations of the form:

|  |  |
| --- | --- |
| Figure - | Figure - |

## Non-aggregated mode

Cost functions and constraints are the same as earlier.

But Lagrange theorem is split in two:

|  |  |
| --- | --- |
| Subject to: | Subject to: |

Find and dependencies on and . Determine s from constraint equations.

# B1) Reinforcement Learning – stochastic approach

Given a system:

We want to find an input that minimize/maximise a cost function. Notice the noise which makes the state transitions stochastic.

Policy – sequence of actions:

The sequence of inputs that we wish to find is called the policy .

The cost function following a policy is given by:

Where the function is the reward function.

Wish to find optimal policy:

**Markov decision processes:**

This RL method uses a model, given a system, the Markov decision process describing the transitions probabilities:



We iteratively approximate the optimal cost function by using the bellman equation.

Bellman can be written into a recursive manner, suitable for solving via a computer program. The Bellman equation is a function for computing the cost/value function, DP is a solution method to find a solution to the bellman.

DP methods can be used to approximate the cost function recursively:



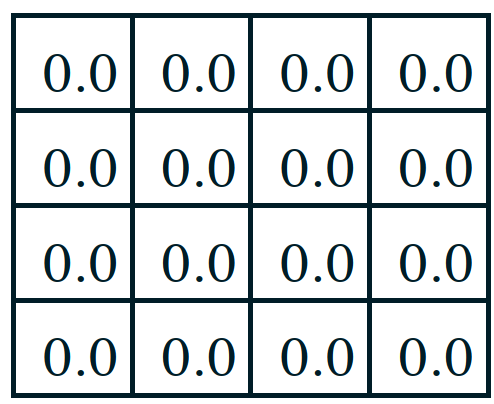
Such that the cost function depends on the reward achieved for being in state i using input u + discounted sum of all possible states and their cost function. Notice and .

Given any initial conditions for (we initialize Q values to 0) the above iterative process will converge to for all states i.

A picture containing diagram

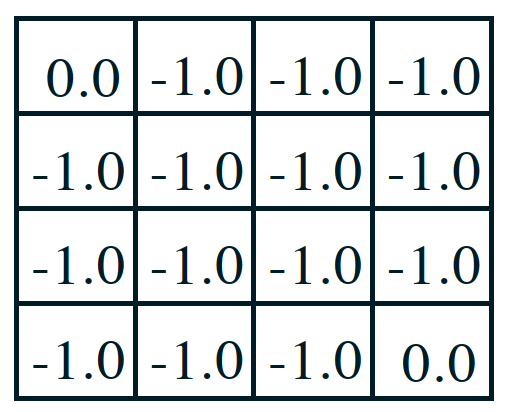
Description automatically generated

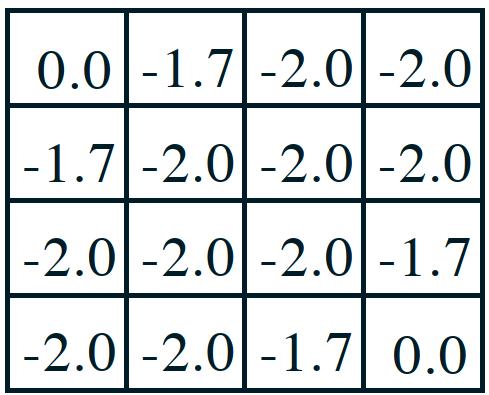
Initialize value function for all states:

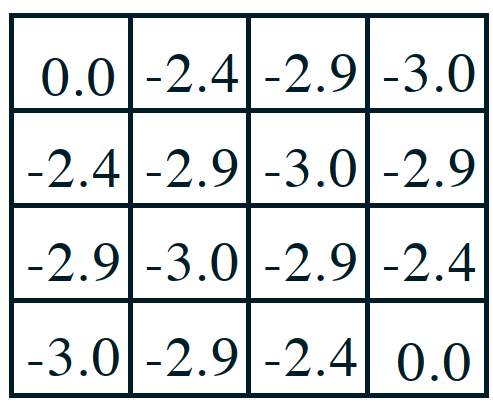


The value function for the problem are written as:

Intuition wise it could look like J(7), would be larger than J(11) since J(11) has a 0. But let’s write up the values in the matrix:







# B2) Model-free Reinforcement Learning – temporal difference, epsilon-greedy exploration, Q-learning

Used to use MC - Don’t need to know Markov decision process transition probabilities!

## Temporal difference

Update value or Q-function some Temporal difference target

is updated towards estimated return aka. Target.

Q-learning:

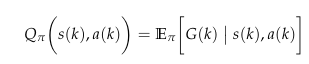
Instead of evaluating we now wish to estimate the expected return of taking an action from a state.

Convergence assumptions:

* Each state and action are visited an infinite amount of times
* and:

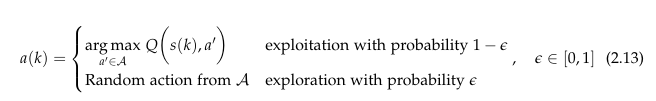
Then:

## Epsilon Greedy policy



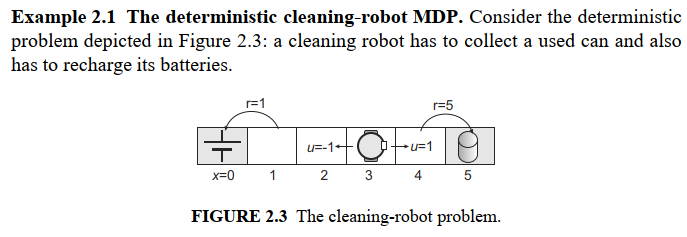
Q might have converged to follow some policy. But if the environment changes continuous exploration is wanted.

From our semester project:



Ensures that all actions are tried an infinite amount of times at from every state as

**Example 2.1 [BBDU]**

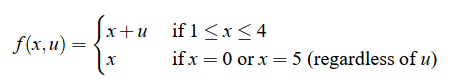


Discrete state space:

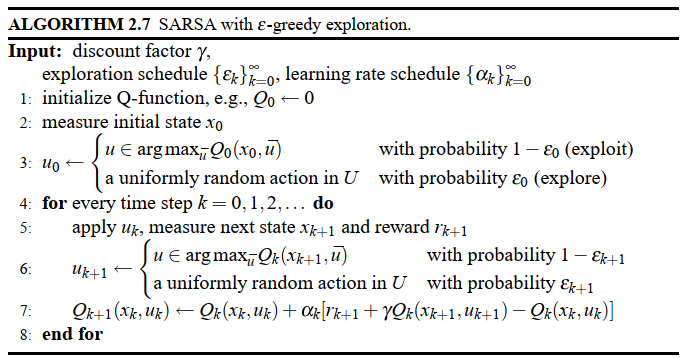
The robot can move left or right:

Reward function is defined as:

Transition function is defined as:

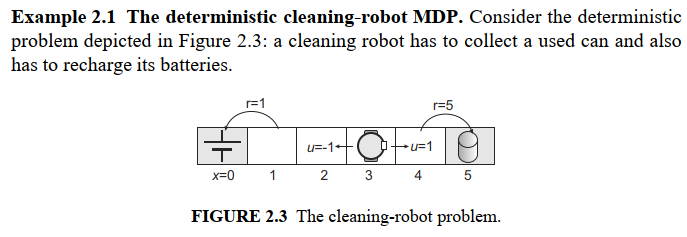


Implement SARSA Algorithm to system:

****

After 10000 iterations the action value function is:

|  |  |
| --- | --- |
|  |  |
|  | 0 | 0 |
|  | 1 | 0.1817 |
|  | 0.4992 | 1.2006 |
|  | 0.2848 | 2.4917 |
|  | 1.2251 | 5 |
|  | 0 | 0 |



**Q(4,-1), Q(4,1):**

****

**One can also use Boltzmann explorations, where there is also a balance between exploration and exploitations like in epsilon greedy, but we do not explore randomly, we instead explore actions/states, that are likely to be candidates to high values. The concept is low value states are unlikely to suddenly be extremely good.**

# C1) Passivity for two systems in negative feedback loop

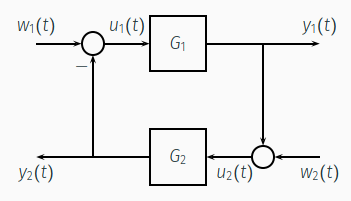
A system is said to be dissipative if:

is a storage function – a measure for the stored energy in the system.

A special type of dissipativity is **Passivity:** choose supply rate

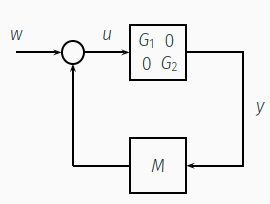
## Dissipasivity of interconnected systems

**Definition of passivity for two systems in negative feedback loop**



The negative feedback connection of two passive systems is passive

Consider:



**Proof:**

Because the system must be dissipative the following inequality must hold:

To check for passivity

Right side becomes for n=1:

Results in:

**Connection to stability:**

**Lemma 5.5**

If a system is **passive** with a positive definite storage function, then the origin of   is Lyapunov stable for .

Furthermore:

**Lemma 5.6**

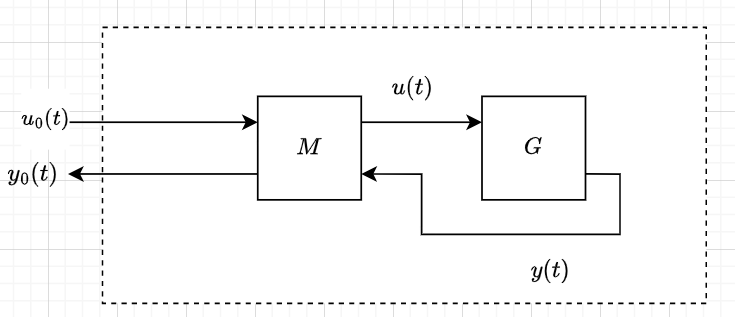
The origin of   is asymptotically stable if:

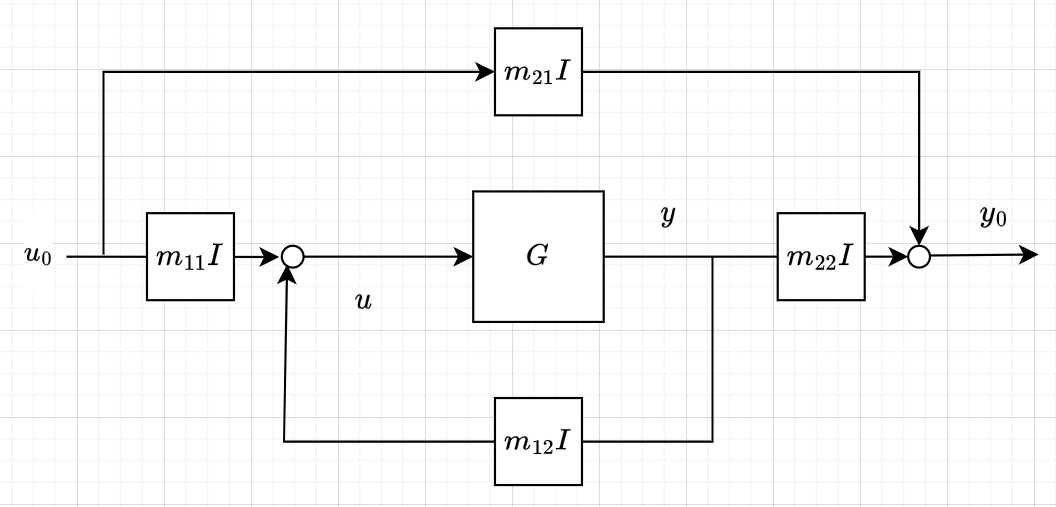
* Strictly passive *or*
* Output strictly passive and zero state observable.

If the storage function is radially unbound then the origin will be globally asymptotically stable

# C2): Non-passive systems exercise

Use -matrix to create so much passivity in controller such that excess than be used to passivate plant.





Consider the negative feedback system shown in C1

Then the two systems are individually passive:

And

Then which results in

**Exercise**

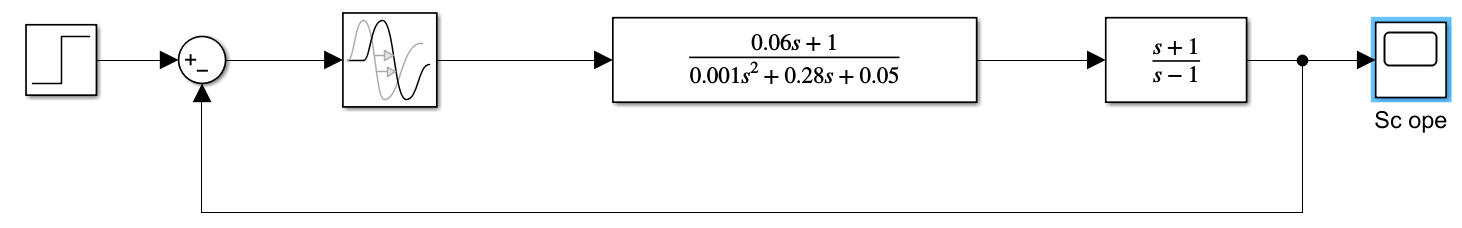
Exercise 2 and 3; explain why the simulations behave as observed.

A Human controller is designed as:

A delayed system can not be passive.

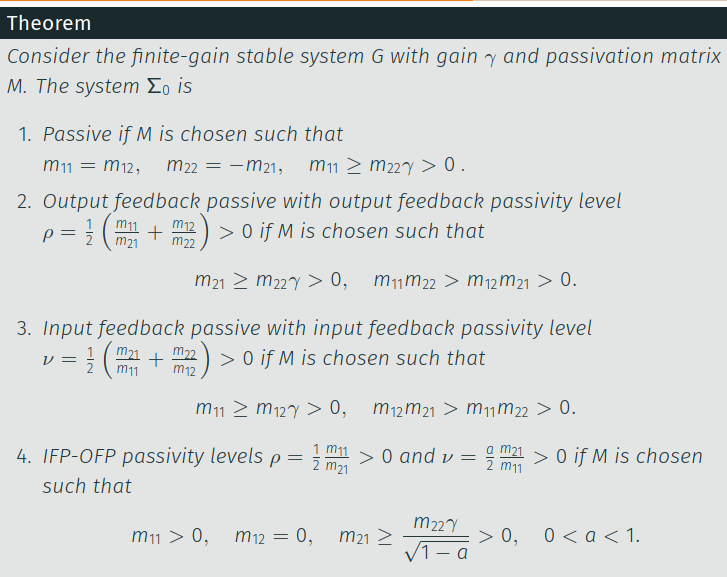
Is used to control

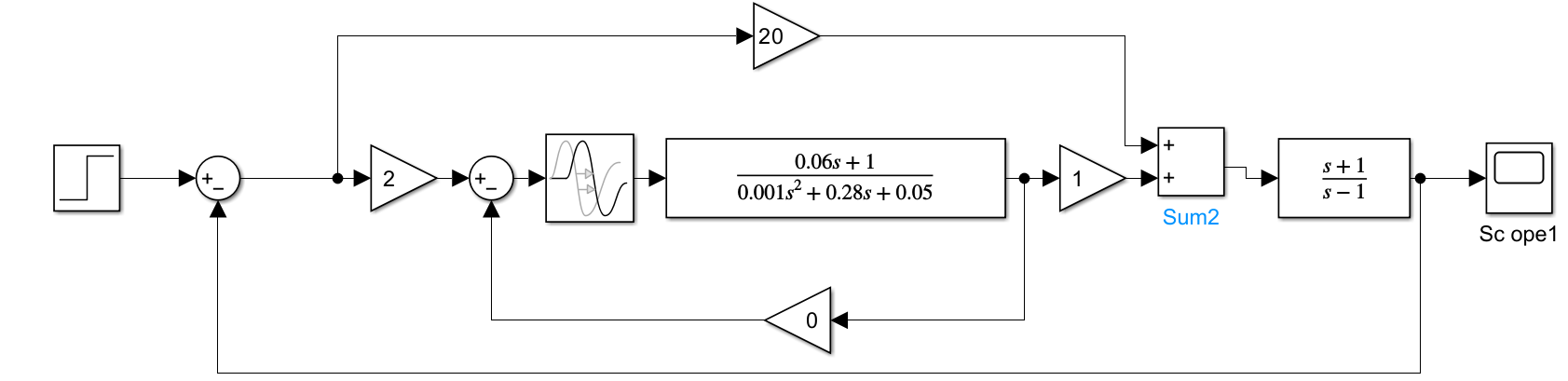
Non-passive controller and non-passive system!



We wish to make the controller so passive that it makes the whole system passive!

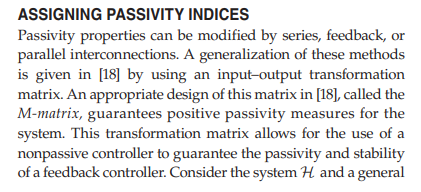
The methods for selecting M indices relies on the condition of your G and H, if the plant H is passive then M can be selected to passivate G, if H is not passive, then we can choose M to let G have excess passivity, which can be used to passivate H.





Results





# Principle of Markov Chain Monte Carlo techniques

*Explain how it is useful for sampling from a posterior as done in exercise 1.1 and exercise 1.2 for lecture 10.*

Goal with Metropolis Hasting algorithm: Wish to make an approximation of a distribution.

Collection of data

Have a model which depends on a parameter . We wish to estimate the distribution of from data .

We have prior model knowledge of the parameter giving us some belief of the pdf of called prior.

Bayes theorem:

Difficult to obtain the posterior

Markov Chain Monte Carlo is a technique to approximate *SAMPLES* from the posterior.  
Can be used to approximate the distribution – could be done by histogram.

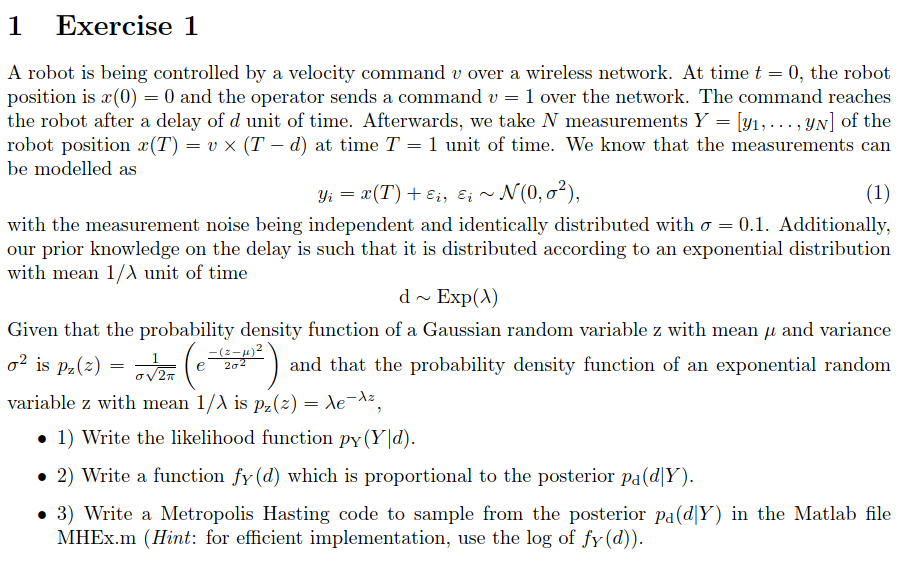
## D1) Metropolis Hastings algorithm - Explain how it is useful for sampling from a posterior

The idea of MCMC is to use a Markov Chain with a *desired(unknown)* stationary distribution and then sample it enough times. We do not actually know the desired distribution, but we know a proportional distribution. We have that:

The Metropolis Hastings can draw samples from a desired distribution provided we have a function proportional to the desired function.

The function we are looking for is then:

### Exercise 1



**The likelihood function**

Measurements are independent ->

is gaussian distributed due to addition of gaussian noise. Therefor mean of is

**Find prior and write function , which is proportional to the posterior**

Prior knowledge of delay is :

We can now determine the proportional function

First terms can be moved outside as long as we remember to raise it to the power of

The Multiplying an exponential function times is the same as summing the exponent times

Remove proportional terms and still holds:

Taking the natural logarithm:

**Algorithm:**

1. Choose arbitrary initial
2. Sample value from proposal density (random walk, symmetrical – normal distribution).
3. Sampled value is accepted or rejected based on the condition:

– this means that the above ratio is ALWAYS accepted if above 1 and SOMETIMES accepted if below 1.

Notice for symetrix Q, they cancel.

Taking the log transforms the division into subtraction for efficiency reasons!

***Burn-in in beginning of transient phase means discarding yearly samples before stationary Markov occurs!***

## D2) Gibbs sampling - Explain how it can be used to sample from a posterior.

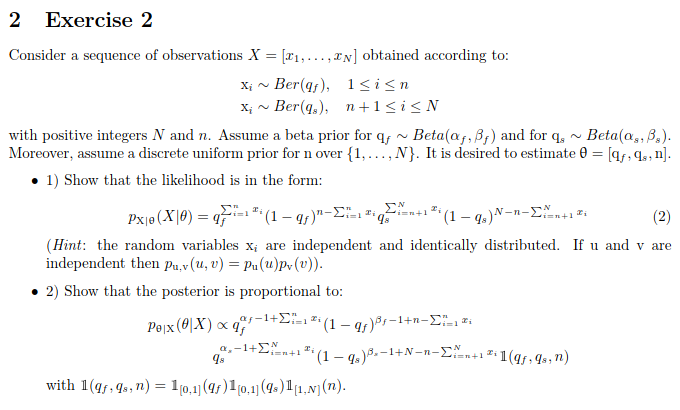
Gibbs sampling is a special case of Metropolis Hastings.

If we consider the case where the posterior is a join probability distribution, like the case for multiple model parameters:

Then sampling from it can be very difficult, since the likelihood and posterior becomes complex.

Instead we try to find the conditionals, and sample from these.

As with Hastings, the idea is to sample these conditionals enough, that the mean gives the true value.



Likelihood function and Posterior are nasty – infeasible to sample from!:

**Likelihood function**

**Posterior**

BUT we can write *the three conditional probabilities and use GIBBS SAMPLING!*:

**Algorithm:**

1. Assume arbitrary and .
2. Draw from each conditional distribution
3. Due to proportionality the found mean is the same as the mean of the wished distribution.

Graphical user interface

Description automatically generated with medium confidence

M = 10 (samples drawn from conditional distributions)

Chart

Description automatically generated

M = 50 (samples drawn from conditional distributions)

Chart

Description automatically generated

M = 500 (samples drawn from conditional distributions)

# D3) Explain how the Conditional Particle Filter with Ancestor Sampling can be used as a Markov Chain Monte Carlo algorithm.

Explain how it can be used for system identification.

Given a stochastic system and a sequence of measurements

* We seek to find the distribution of:

We are primarily interested in parameter estimation. The above is a joint probability, for which we want to use Gibbs sampling to acquire the conditional distributions:

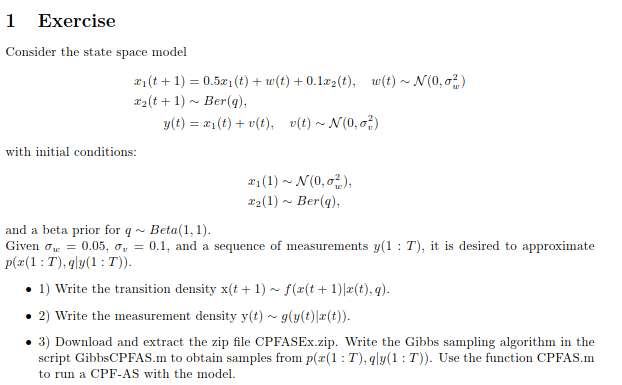
Being a MCMC method, we require a Markov chain for which we can let the stationary distribution go towards a desired distribution. Thereby, if given the m’th sample of the Markov chain we want to find . This Markov chain can be acquired by using CPF-AS.



Text

Description automatically generated

## Exercise 1

****

**Transition density**

**Write down measurement density**

