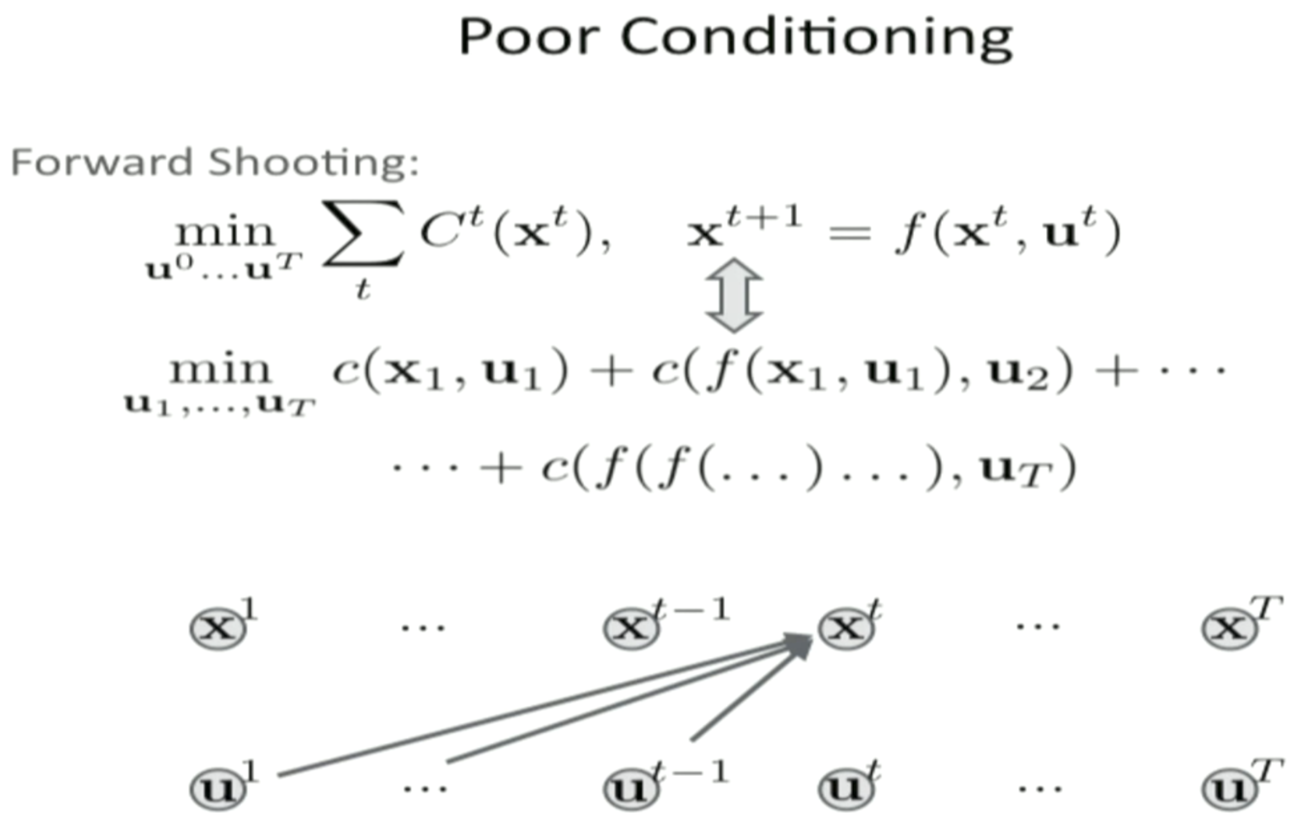
direct collocation

- shooting methods have the downside of poor conditioning. changing u_1 has a very different effect than changing u_T.



Poor Conditioning

Forward Shooting:

$$\min_{\mathbf{u}^0 \ldots \mathbf{u}^T} \sum_t C^t(\mathbf{x}^t), \qquad \mathbf{x}^{t+1} = f(\mathbf{x}^t, \mathbf{u}^t)$$

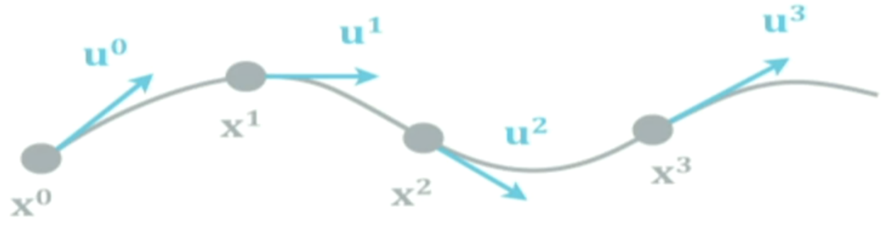
$$\Updownarrow$$

$$\min_{\mathbf{u}_1,\ldots,\mathbf{u}_T} c(\mathbf{x}_1, \mathbf{u}_1) + c(f(\mathbf{x}_1, \mathbf{u}_1), \mathbf{u}_2) + \cdots$$
$$\cdots + c(f(f(\ldots)\ldots), \mathbf{u}_T)$$

- they also have the big downside that you could fall into local optima! the non-linear dynamics iLQR we talked about can be very much non-convex. this is a big one we didn't mention before. this makes it sensitive to initial guess. we could get around this by initial guess from demonstrations, but its not always possible depending on the domain.
- With direct collocation, we only deal in state space and then later compute actions with an inverse dynamics function (f^{-1}).

Direct Collocation:

$$\min_{\mathbf{x}^0 \ldots \mathbf{x}^T} \sum_t C^t(\mathbf{x}^t), \;\; st \;\; f^{-1}(\mathbf{x}^t, \mathbf{x}^{t+1}) = \mathbf{u}^t \in \mathcal{U}$$



- the inverse dynamics function says what action to apply to get from x_t to x_{t+1}. Notice that we have an explicit constraint that all consecutive pairs of states must be reachable with a possible action.
- just like we learnt the dynamics model, we can learn an inverse dynamics model from data (if we don't already know the physics of the system. if we do we can just derive it analytically)
- while doing this approach, we have to realize that the inverse dynamics function isn't always possible for all x_t to x_{t+1} unless we imposed a hard constraint on the above optimization problem. for example it may not be possible to get from state a to state b with one action while using soft constraints.
- With collocation, we don't have the same conditioning problem as before, our states only have pairwise dependencies and no forward integration instability where errors with dynamics could accumulate over time.
- The big win is that is that we are also less prone to local minima and the satisfaction of dynamics is explicit so we can make it soft or hard (hard feasible region vs energy based soft regions)
- A summary of the differences between forward shooting and direct collocation:

Forward Shooting:

$$\min_{\mathbf{u}^0 \ldots \mathbf{u}^T} \sum_t C^t(\mathbf{x}^t), \quad \mathbf{x}^{t+1} = f(\mathbf{x}^t, \mathbf{u}^t)$$

- Optimize over controls
- State trajectory is implicit
- Dynamics is an implicit constraint (always satisfied)

Direct Collocation:

$$\min_{\mathbf{x}^0 \ldots \mathbf{x}^T} \sum_t C^t(\mathbf{x}^t), \;\; st \;\; f^{-1}(\mathbf{x}^t, \mathbf{x}^{t+1}) = \mathbf{u}^t \in \mathcal{U}$$

- Optimize over states
- Controls and forces are implicit
- Dynamics is an explicit constraint (can be soft)

- both shooting and collocation can be applied to control of movement without contact. eg. flying, driving, swimming, collision free path finding etc.
- when contact is involved, its hard to apply either shooting or collocation, because of discontinuous jumps caused by the contact forces.
- also, we would have no gradient information from inactive contacts that might become active in one or more of the trajectories under consideration, so can't anticipate being able to apply forces.
- One of the advantages of soft constraint dynamics is that they can jump to the target soon, thereby enabling the rest of the optimization to jump out of local minima. Desired behavior with contacts (eg should not slide) are incorporated into the optimization cost function.
- the above challenges bring us to contact invariant optimization, which open ai is working on, says igor.
- Igor shows videos of some very natural looking, contact related tasks, but does not present details on the methods that were used for it.
- While trajectory optimization is an automatic and general approach, it seems unlikely that we humans do that and also there is no reuse of the optimization → we run optimization every time. this motivates

learning policies. the ideas about learning policies with direct collocation are not very different from the ones we saw in guided policy search (which we saw with forward shooting)

- Igor demos some models trained in simulation with some noise injected into the model and then runs them in the real world → this works better than assuming that you have a perfect model in simulation.