Hamilton-Jacobi Equations for Optimal Control and Reachability

Ian Mitchell
Department of Computer Science
The University of British Columbia

Joint work with
Alex Bayen, Meeko Oishi & Claire Tomlin (Stanford)
Jon Sprinkle, Mike Eklund & Shankar Sastry (Berkeley)

research supported by
National Science and Engineering Research Council of Canada
DARPA Software Enabled Control Project
Outline

• **Dynamic programming for discrete time optimal control**

• **Hamilton-Jacobi equation for continuous time optimal control**
  – Formal derivation
  – Time-dependent vs static
  – Viscosity solution concepts

• **Hamilton-Jacobi equation for reachability**
  – Characterization of backwards reach tube
  – Examples and applications
  – Dimensional reduction through projection
Dynamic Programming in Discrete Time

• Consider finite horizon objective with $\alpha = 1$ (no discount)

Let $x(\cdot)$ solve $x(t + 1) = f(x(t), u(t))$ and $x(t_0) = x_0$

$$J(t_0, x_0, u(\cdot)) = \sum_{t=t_0}^{t_f-1} g(x(t), u(t)) + g_f(x(t_f))$$

$$= g(x(t_0), u(t_0)) + J(t_0 + 1, x(t_0 + 1), u(\cdot))$$

$$= g(x(t_0), u(t_0)) + J(t_0 + 1, f(x(t_0), u(t_0)), u(\cdot))$$

$$J(t_f, x(t_f), u(\cdot)) = g_f(x(t_f))$$

• So given $u(\cdot)$ we can solve inductively backwards in time for objective $J(t, x, u(\cdot))$, starting at $t = t_f$
  – Called dynamic programming (DP)
Optimal Control via Dynamic Programming

• DP can also be applied to the value function
  – Second step works because $u(t_0)$ can be chosen independently of $u(t)$ for $t > t_0$

$$V(t_0, x_0) = \min_{u(\cdot)} \sum_{t=t_0}^{t_f-1} g(x(t), u(t)) + g_f(x(t_f))$$

$$= \min_{u(\cdot)} (g(x(t_0), u(t_0)) + J(t_0 + 1, x(t_0 + 1), u(\cdot)))$$

$$= \min_{u} g(x(t_0), u) + V(t_0 + 1, x(t_0 + 1))$$

$$= \min_{u} g(x(t_0), u) + V(t_0 + 1, f(x(t_0), u(t_0)))$$

$$V(t_f, x(t_f)) = g_f(x(t_f))$$
Optimal Control via Dynamic Programming

• Optimal control signal $u^*(\varsigma)$

$$u^*(t, x(t)) = \arg \min_u g(x(t), u) + V(t+1, f(x(t), u))$$

• Observe update equation

$$\Delta V(t, x) = V(t + 1, x(t + 1)) - V(t, x)$$

$$= - \min_u g(x, l, u)$$

• Can be extended (with appropriate care) to
  – other objectives
  – probabilistic models
  – adversarial models
Discrete Time LQR

• Stephen Boyd’s class at Stanford: EE363 Linear Dynamical Systems:
  http://www.stanford.edu/class/ee363/

• Discrete LQR: lecture notes 1
  – pages 1-1 to 1-4 (from 2005): LQR background
  – pages 1-13 to 1-16 (from 2005): Dynamic programming sol’n
  – pages 1-19 to 1-23 (from 2005): HJ equation for LQR
Continuous Time LQR

• Stephen Boyd’s class at Stanford: EE363 Linear Dynamical Systems:
  http://www.stanford.edu/class/ee363/

• Continuous LQR: lecture notes 4
  – pages 4-1 to 4-9 (from 2005): derivation of HJ (and Riccati differential equation) via discretization
Outline

• Dynamic programming for discrete time optimal control

• Hamilton-Jacobi equation for continuous time optimal control
  – Formal derivation
  – Time-dependent vs static
  – Viscosity solution concepts

• Hamilton-Jacobi equation for reachability
  – Characterization of backwards reach tube
  – Examples and applications
  – Dimensional reduction through projection
Dynamic Programming Principle

$$\vartheta(x) = \min_{y \in N(x)} [\vartheta(y) + c(x \rightarrow y)]$$

- **Target set problem**
  - Value function $\vartheta(x)$ is “cost to go” from $x$ to the nearest target
- **Value $\vartheta(x)$** at a point $x$ is the minimum over all points $y$ in the neighborhood $N(x)$ of the sum of
  - the value $\vartheta(y)$ at point $y$
  - the cost $c(x \rightarrow y)$ to travel from $x$ to $y$
- **Dynamic programming applies if**
  - Costs are additive
  - Subsets of feasible paths are themselves feasible
  - Concatenations of feasible paths are feasible
Continuous Dynamic Programming

- Discrete dynamic programming principle
  \[ \vartheta(x) = \min_{y \in N(x)} [\vartheta(y) + c(x \rightarrow y)] \]

- Continuous DPP for path \( p(\cdot) \)
  \[ \vartheta(p(s)) = \min_{p(\cdot)} \left[ \vartheta(p(s + \Delta s)) + \int_{s}^{s+\Delta s} c(p(\sigma)) d\sigma \right] \]

- Rearrange (take min over \( p(\cdot) \) everywhere)
  \[ \frac{\vartheta(p(s)) - \vartheta(p(s + \Delta s))}{\Delta s} - \frac{\int_{s}^{s+\Delta s} c(p(\sigma)) d\sigma}{\Delta s} = 0 \]

- Take limit \( \Delta s \rightarrow 0 \)
  \[ -\frac{d\vartheta(p(s))}{ds} - c(p(s)) = 0 \]
Static Hamilton-Jacobi PDE
Hamilton-Jacobi Flavours

• Stationary (static/time-independent) Hamilton-Jacobi used for target based cost to go and time to reach problems

$$H(x, D_x \vartheta(x)) = 0 \quad \| \nabla \vartheta(x) \| = c(x)$$

  – PDE coupled to boundary conditions
  – Solution may be discontinuous

• Time-dependent Hamilton-Jacobi used for dynamic implicit surfaces and finite horizon optimal control / differential games

$$D_t \phi(x, t) + H(x, D_x \phi(x, t)) = 0$$

  – PDE coupled to initial/terminal and possibly boundary conditions
  – Solution continuous but not necessarily differentiable

• Other versions exist
  – Discounted and/or infinite horizon
Time-Dependent Hamilton-Jacobi PDE

- Start with dynamic programming equation for value function $\phi(t, x)$

$$
\phi(t, x(t)) = \min_{u(\cdot)} \left[ \int_t^{t+\Delta t} g(x(s), u(s)) \, ds + \phi(t + \Delta t, x(t + \Delta t)) \right]
$$

- Rearrange, divide by $\Delta t$

$$
\min_{u(\cdot)} \left( \frac{1}{\Delta t} \right) \left[ \phi(t + \Delta t, x(t + \Delta t)) - \phi(t, x(t)) \right] + \int_t^{t+\Delta t} g(x(s), u(s)) \, ds = 0
$$

- After limit $\Delta t \to 0$

$$
\min_{u(\cdot)} \left[ \frac{d\phi(t, x(t))}{dt} + g(x(t), u(t)) \right] = 0
$$
Time-Dependent Hamilton-Jacobi PDE

- After limit $\Delta t \to 0$

$$\min_{u(\cdot)} \left[ \frac{d\phi(t, x(t))}{dt} + g(x(t), u(t)) \right] = 0$$

- Apply chain rule

$$\min_{u(\cdot)} \left[ \frac{\partial \phi(t, x)}{\partial t} \frac{\partial \phi(t, x)}{\partial x} \frac{dx(t)}{dt} + g(x, u(t)) \right] = 0$$

$$\min_{u(\cdot)} \left[ \frac{\partial \phi(t, x)}{\partial t} \frac{\partial \phi(t, x)}{\partial x} f(x, u(t)) + g(x, u(t)) \right] = 0$$

- Observe that result depends only on $u(t)$, not $u(\cdot)$

$$D_t \phi(t, x) + H(x, D_x \phi(t, x)) = 0$$

where

$$H(x, p) = \min_u [D_x \phi(t, x) \cdot f(x, u) + g(x, u)]$$
Time-Dependent Hamilton-Jacobi PDE

- Finite horizon problem value function defined by

\[
\phi(t_0, x_0) = \min_{u(\cdot) \in \mathcal{U}} \int_{t_0}^{t_f} g(x(t), u(t)) + g_f(x(t_f))
\]

- From \( t = t_f \), we get PDE with terminal conditions

\[
D_t \phi(t, x) + H(x, D_x \phi(t, x)) = 0
\]

where

\[
H(x, p) = \min_u [D_x \phi(t, x) \cdot f(x, u) + g(x, u)]
\]

\[
\phi(t_f, x) = g_f(x(t_f))
\]

- A very informal derivation of the time-dependent Hamilton-Jacobi equation
- Rigourous derivations must account for \( \phi(t, x) \) not differentiable, the optimal \( u(\phi) \) may not exist, ...
- Resolution: viscosity solutions
Viscosity Solutions

• Hamilton-Jacobi PDE for optimal control derived by Bellman in 1950s, but in general there is no classical solution
  – Even if dynamics $f$ and cost functions $g$ and $g_t$ are smooth, solution may not be differentiable
• Early attempt to define a weak solution: “vanishing viscosity”
  – Take limit as $\epsilon \to 0$ of solution to
    \[ D_t \phi(t, x) + H(x, D_x \phi(t, x)) = \epsilon D_x^2 \phi(t, x) \]
  – Unfortunately, limit may not exist
• Crandall & Lions (1983) propose “viscosity solution”
  – Under reasonable conditions there exists a unique solution
  – If there exists a classical solution, then it is the same as the viscosity solution
  – If there exists a vanishing viscosity solution, then it is the same as the viscosity solution
• Not the only possible weak solution, but it is the right one for control and reachability
Viscosity Solution Definition

• Original definition no longer used
• More common definition from Crandall, Evans & Lions (1984)
  – Solution $\phi(t, x)$ satisfies terminal conditions
  – For each smooth test function $\psi(t, x)$

\[
\text{If } \phi(t, x) - \psi(t, x) \text{ has a local maximum, then } \\
D_t \psi(t, x) + H(x, D_x \psi(t, x)) \geq 0
\]

\[
\text{If } \phi(t, x) - \psi(t, x) \text{ has a local minimum, then } \\
D_t \psi(t, x) + H(x, D_x \psi(t, x)) \leq 0
\]

• Solution satisfies a comparison principle
  – If $\phi$ and $\psi$ solve the same HJ PDE where $\phi \geq \psi$ on the boundary (including initial or terminal times), then $\phi \geq \psi$ everywhere
  – Comparison principles are now the preferred way of characterizing viscosity solutions for PDEs
Viscosity Solution Citations

• Since extended to
  – Discounted infinite horizon
  – Static (for minimum cost to go)
  – Two player zero sum games
  – Degenerate elliptic (for SDE dynamics)

• Citations:
  – Viscosity Solutions & Applications Springer’s Lecture Notes in Mathematics (1995), featuring Bardi, Crandall, Evans, Soner & Souganidis (Capuzzo-Dolcetta & Lions eds)
Outline

• Dynamic programming for discrete time optimal control

• Hamilton-Jacobi equation for continuous time optimal control
  – Formal derivation
  – Time-dependent vs static
  – Viscosity solution concepts

• Hamilton-Jacobi equation for reachability
  – Characterization of backwards reach tube
  – Examples and applications
  – Dimensional reduction through projection
Calculating Reach Sets

- Two primary challenges
  - How to represent set of reachable states
  - How to evolve set according to dynamics
- Discrete systems $x_{k+1} = \delta(x_k)$
  - Enumerate trajectories and states
  - Efficient representations: Binary Decision Diagrams
- Continuous systems $\frac{dx}{dt} = f(x)$?
Implicit Surface Functions

- Set $G(t)$ is defined implicitly by an isosurface of a scalar function $\phi(x,t)$, with several benefits
  - State space dimension does not matter conceptually
  - Surfaces automatically merge and/or separate
  - Geometric quantities are easy to calculate

- Set must have an interior
  - Examples (and counter-examples) shown on board

$$\phi : \mathbb{R}^n \times \mathbb{R} \rightarrow \mathbb{R} \quad G(t) = \{ x \in \mathbb{R}^n \mid \phi(x, t) \leq 0 \}$$
Reachable Set as Optimal Control

- Represent the target set as an implicit surface function
  \[ T = G(0) = \{ x \mid h(x) \leq 0 \} \]

- Solve an optimal control problem with target set implicit surface function as the terminal cost, and zero running cost

  initial conditions: \[ \xi_f(t; x, t, a(\cdot), b(\cdot)) = x \]
  dynamics: \[ \dot{\xi}_f(s; x, t, a(\cdot), b(\cdot)) = f(x, a(s), b(s)) \]
  running cost: \[ g(x, a, b) = 0 \]
  terminal cost: \[ g_f(x) = h(x) \]

- Resulting value function is an implicit surface function for the backward reach set
  \[ V(x, t) = \phi(x, t) = \inf_{\gamma[a(\cdot)](\cdot)} \sup_{\alpha(\cdot)} \left( h \left[ \xi_f(0; x, t, a(\cdot), \gamma[a(\cdot)](\cdot)) \right] \right) \]
  \[ G(t) = \{ x \mid \phi(x, t) \leq 0 \} \]
Game of Two Identical Vehicles

• Classical collision avoidance example
  – Collision occurs if vehicles get within five units of one another
  – Evader chooses turn rate $|a| \leq 1$ to avoid collision
  – Pursuer chooses turn rate $|b| \leq 1$ to cause collision
  – Fixed equal velocity $v_e = v_p = 5$

\[
\frac{d}{dt} \begin{bmatrix} x_p \\ y_p \\ \theta_p \end{bmatrix} = \begin{bmatrix} v_p \cos \theta_p \\ v_p \sin \theta_p \\ b \end{bmatrix}
\]

evader aircraft (control) | pursuer aircraft (disturbance)
Collision Avoidance Computation

- Work in relative coordinates with evader fixed at origin
  - State variables are now relative planar location \((x, y)\) and relative heading \(\psi\)

\[
\frac{d}{dt} \begin{bmatrix} x \\ y \\ \psi \end{bmatrix} = \begin{bmatrix} -v_e + v_p \cos \psi - ay \\ v_p \sin \psi - ax \\ b - a \end{bmatrix} = f(z, a, b)
\]

target set description
\[h(x) = \sqrt{x^2 + y^2} - 5\]

evader aircraft (control)    pursuer aircraft (disturbance)
Evolving Reachable Sets

- Modified Hamilton-Jacobi partial differential equation

\[ D_t \phi(z, t) + \min [0, H(z, D_z \phi(z, t))] = 0 \]

with Hamiltonian: \( H(z, p) = \max_{a \in \Lambda} \min_{b \in B} f(z, a, b) \cdot p \)

and terminal conditions: \( \phi(z, 0) = h(z) \)

where \( G(0) = \{ z \in \mathbb{R}^n | h(z) \leq 0 \} \)

and \( \dot{z} = f(z, a, b) \)

growth of reachable set

final reachable set

Ian Mitchell (UBC Computer Science)
Solving a Differential Game

- Terminal cost differential game for trajectories $\xi_f(c; x, t, a(c), b(c))$

$$\phi(x, t) = \inf_{\gamma[a(.)](\cdot)} \sup_{a(\cdot)} h \left[ \xi_f(0; x, t, a(.), \gamma[a(.)](\cdot)) \right]$$

where

$$\begin{cases} 
\xi_f(t; x, t, a(.), b(.)) = x \\
\dot{\xi}_f((s; x, t, a(.), b(.)) = f(x, a(s), b(s)) 
\end{cases}$$

- Value function solution $\phi(x,t)$ given by viscosity solution to basic Hamilton-Jacobi equation
  
  - [Evans & Souganidis, 1984]
  
  $$D_t \phi(x, t) + H(x, D_x \phi(x, t)) = 0$$

  where

  $$\begin{cases} 
  H(x, p) = \max_{a \in A} \min_{b \in B} p^T f(x, a, b) \\
  \phi(x, 0) = h(x) 
  \end{cases}$$
Modification for Optimal Stopping Time

• How to keep trajectories from passing through $G(0)$?
  – Augment disturbance input
    \[
    \tilde{b} = \begin{bmatrix} b & b \end{bmatrix} \quad \text{where} \quad b : [t, 0] \to [0, 1]
    \]
    \[
    \tilde{f}(x, a, \tilde{b}) = b f(x, a, b)
    \]
  – Augmented Hamilton-Jacobi equation solves for reachable set
    \[
    D_t \phi(x, t) + \tilde{H}(x, D_x \phi(x, t)) = 0 \quad \text{where}
    \begin{cases}
    \tilde{H}(x, p) = \max_{a \in \mathcal{A}} \min_{b \in \mathcal{B}} p^T \tilde{f}(x, a, \tilde{b}) \\
    \phi(x, 0) = h(x)
    \end{cases}
    \]
  – Augmented Hamiltonian is equivalent to modified Hamiltonian
    \[
    \tilde{H}(x, p) = \max_{a \in \mathcal{A}} \min_{\tilde{b} \in \mathcal{B}} p^T \tilde{f}(x, a, \tilde{b})
    \]
    \[
    = \max_{a \in \mathcal{A}} \min_{b \in \mathcal{B}} \min_{\tilde{b} \in [0,1]} b p^T f(x, a, b)
    \]
    \[
    = \min \left[ 0, \max_{a \in \mathcal{A}} \min_{b \in \mathcal{B}} p^T f(x, a, b) \right] = \min [0, H(x, p)]
    \]
Application: Collision Alert for ATC

- Use reachable set to detect potential collisions and warn Air Traffic Control (ATC)
  - Find aircraft pairs in ETMS database whose flight plans intersect
  - Check whether either aircraft is in the other’s collision region
  - If so, examine ETMS data to see if aircraft path is deviated
  - One hour sample in Oakland center’s airspace—
    - 1590 pairs, 1555 no conflict, 25 detected conflicts, 2 false alerts
Application: Synthesizing Safe Controllers

- By construction, on the boundary of the unsafe set there exists a control to keep trajectories safe
  - Filter potentially unsafe controls to ensure safety

\[
\dot{x} = f(x, u, d)
\]

\[
\forall x \in \partial G(t), \exists u \in U, \forall d \in D
\]

\[
n(x) \cdot f(x, u, d) \geq 0
\]
Synthesizing Safe Controls (No Safety)

- Use reachable sets to guarantee safety
- Basic Rules
  - Pursuer: turn to head toward evader
  - Evader: turn to head east
- No filtering of evader input
Synthesizing Safe Controls (Success)

- Use reachable sets to guarantee safety
- Basic Rules
  - Pursuer: turn to head toward evader
  - Evader: turn to head east
- Evader’s input is filtered to guarantee that pursuer does not enter the reachable set
Synthesizing Safe Controls (Failure)

- Use reachable sets to guarantee safety
- Basic Rules
  - Pursuer: turn to head toward evader
  - Evader: turn to head east
- Evader’s input is filtered, but pursuer is already inside reachable set, so collision cannot be avoided
Acoustic Capture

• Modified version of homicidal chauffeur from [Cardaliaguet, Quincampoix & Saint-Pierre, 1999]
  – Pursuer is faster with limited turn radius but fast rotation
  – Evader can move any direction, but speed is lowered near pursuer

• Also solved in relative coordinates

continuous system dynamics

\[ \frac{d}{dt} \begin{bmatrix} x \\ y \end{bmatrix} = W_p \begin{bmatrix} 0 \\ -1 \end{bmatrix} + \frac{W_p}{R} \begin{bmatrix} y \\ -x \end{bmatrix} b \]
\[ + 2W_e \min \left( \sqrt{x^2 + y^2}, S \right) a \]

\[ a \in \mathbb{R}^2, \|a\| \leq 1 \]
\[ b \in [-1, +1] \]
\[ W_p, W_e, R, S \text{ constant} \]
Application: Glideslope Recapture

(a) safe-set of operation relative to the desired point of landing on the virtual runway (f)
(b) vector-off maneuver requested
(c) command to land (if possible) is given
(d) aircraft will continue to vector-off (if landing is unsafe) or will issue commands to recapture the glideslope at some point (e)
Simple Model, Use Some Tricks

• Aircraft acts like a kinematic cart in longitudinal and lateral dynamics

• Reduce 5D state space to two 2D calculations
  – Separate \((x,y,\theta)\) and \((x,z,\psi)\) calculations
  – Swap integration variable \(t\) to \(x\)

• Variable resolution grid (two methods)


\[
\begin{align*}
\begin{bmatrix}
x \\
y \\
z \\
\theta \\
\psi \\
\end{bmatrix} &= \begin{bmatrix}
x \cos \psi \cos \theta \\
v \sin \psi \\
v \sin \theta \\
u_\theta \\
u_\psi \\
\end{bmatrix} \\
d \begin{bmatrix}
x \\
y \\
z \\
\theta \\
\psi \\
\end{bmatrix}
\end{align*}
\]

- \(x\) distance along runway
- \(y\) distance across runway
- \(z\) height
- \(\theta\) yaw
- \(\psi\) roll

\[
\begin{align*}
u_\theta & \in U_\theta & u_\psi & \in U_\psi \\
\theta & \in [\theta_{\text{min}}, \theta_{\text{max}}] & v & \text{fixed} \\
\psi & \in [\psi_{\text{min}}, \psi_{\text{max}}] \\
\end{align*}
\]
Implementation and Results

- ACC results shown; ISSE results difficult to visualize
- All pieces fit together, grid resolution changes by factor of ten
- ISSE uses nonlinear coordinate transform to avoid the need for separate grids

\[ [x, z, \theta] \text{ projection} \]
Implementation and Results

\[ [0, 1) \] Safe Set from \([0, 1)\) nautical miles

\[ [1, 3) \] Safe Set from \([1, 3)\) nautical miles

\[ [3, 10) \] Safe Set from \([3, 10)\) nautical miles

[x, y, ψ] projection
Implementation

- Use backward reach set to make one lookup table for each projection
  - ~7MB total size
  - Lookup time: ~10ms (~5ms each)
  - Time to generate:
    - ~15 mins for the reach set
    - ~2 hours to compile into an executable (due to compiler issues)
- Additional issues
  - Smooth control laws
  - Decision delay
- Total development time
  - ~2 man months of coding, plus design and research required for safe sets
Demonstration & Results

- Flown on live T-33 aircraft
- Landing on “virtual” runway at a high altitude
- Ground controller gives vector-off and recapture commands
  - 1 successful landing
  - 1 go-around after “unsafe” answer (later verified offline as a correct result)
Projective Overapproximation

- Overapproximate reachable set of high dimensional system as the intersection of reachable sets for lower dimensional projections
  - [Mitchell & Tomlin, 2002]
  - Example: rotation of “sphere” about z-axis
Computing with Projections

• Forward and backward reachable sets for finite automata
  – Projecting into overlapping subsets of the variables, computing with BDDs [Govindaraju, Dill, Hu, Horowitz]

• Forward reachable sets for continuous systems
  – Projecting into 2D subspaces, representation by polygons [Greenstreet & Mitchell]

• Level set algorithms for geometric optics
  – Need multiple arrival time (viscosity solution gives first arrival time), so compute in higher dimensions and project down [Osher, Cheng, Kang, Shim & Tsai]
Hamilton-Jacobi in the Projection

• Consider $x$-$z$ projection represented by level set $\phi_{xz}(x,z,t)$
  – Back projection into 3D yields a cylinder $\phi_{xz}(x,y,z,t)$
• Simple HJ PDE for this cylinder

$$D_t \phi_{xz}(x, y, z, t) + \sum_{i=1}^{3} p_i f_i(x, y, z) = 0 \quad \text{where} \quad \begin{cases} p_1 = D_x \phi_{xz}(x, y, z, t) \\ p_2 = D_y \phi_{xz}(x, y, z, t) \\ p_3 = D_z \phi_{xz}(x, y, z, t) \end{cases}$$

  – But for cylinder parallel to $y$-axis, $p_2 = 0$

$$D_t \phi_{xz}(x, y, z, t) + p_1 f_1(x, y, z) + p_3 f_3(x, y, z) = 0$$

• What value to give free variable $y$ in $f_i(x,y,z)$?
  – Treat it as a disturbance, bounded by the other projections

$$D_t \phi_{xz}(x, y, z, t) + \min_y [p_1 f_1(x, y, z) + p_3 f_3(x, y, z)] = 0$$

• Hamiltonian no longer depends on $y$, so computation can be done entirely in $x$-$z$ space on $\phi_{xz}(x,z,t)$
Projective Collision Avoidance

- Work strictly in relative $x$-$y$ plane
  - Treat relative heading $\psi \in [0, 2\pi]$ as a disturbance input
  - Compute time: 40 seconds in 2D vs 20 minutes in 3D
  - Compare overapproximative prism (mesh) to true set (solid)
Projection Choices

- Poorly chosen projections may lead to large overapproximations
  - Projections need not be along coordinate axes
  - Number of projections is not constrained by number of dimensions