PDE for Finance Notes – Section 1

Notes by Robert V. Kohn, Courant Institute of Mathematical Sciences. For use only in connection with the NYU course PDE for Finance, G63.2706, Spring 1999.

Deterministic optimal control, dynamic programming, and the Hamilton-Jacobi-Bellman equation. This section gives a fast introduction to optimal control via dynamic programming. We mainly follow chapter 10 of Evans, *Partial Differential Equations*, and Section 1 of the chapter by Bardi in *Viscosity Solutions and Applications* (Springer Lecture Notes in Math 1660). Anticipating heavy demand for Evans' book (which only just arrived in the library), I've placed a xerox copy of Chapter 10 in the Green Box on reserve. A standard text on dynamic programming and optimal control, covering many examples (mainly from physical sciences) and many topics not addressed here (such as Pontryagin's maximum principle), is W. Fleming and R. Rishel, *Deterministic and stochastic optimal control.*

****

A typical problem of optimal control is this: we have a system, whose *state* at any time t is described by a vector $x = x(t) \in R^n$. The system evolves in time, and we have the ability to influence its evolution through a vector-valued *control* $\alpha(t) \in R^m$. The evolution of the system is determined by an *ordinary differential equation*

$$\dot{x}(t) = f(x(t), \alpha(t)), \quad x(0) = x_0,$$

and our goal is to choose the function $\alpha(t)$ for $0 < t < T$ so as to minimize some cost or maximize some utility, e.g.

$$\min \int_0^T h(x(s), \alpha(s))\, ds + g(x(T)).$$

The problem is determined by specifying the dynamics f, the initial state x_0, the final time T, the "running cost" h and the final cost g. The problem is solved by finding the optimal control $\alpha(t)$ for $0 < t < T$ and the value of the minimum.

The mathematical literature usually focuses on minimizing cost; the economic literature on maximizing utility. The two problems are mathematically equivalent.

One needs some hypotheses on f to be sure the solution of the ODE defining $x(t)$ exists and is unique. We do not make these explicit since the goal of these notes is to summarize the main ideas without getting caught up in fine points. See Evans for a mathematically careful treatment. Another technical point: it's possible to formulate an optimal control problem that has no solution. If the cost is bounded below, then for any $\epsilon > 0$ there's certainly a control $\alpha_\epsilon(t)$ achieving a cost within ϵ of optimal. But the controls α_ϵ might not converge to a meaningful control as $\epsilon \to 0$. (See Homework 1 for a simple example with this difficulty.) Note however that even if an optimal control doesn't exist, the optimal *cost* is still well-defined.

Dynamic programming is a method for approaching problems like this. The idea is simple: embed the problem you wish to solve in a parametrized *family* of problems, with the property that solving each one makes the next one easy. In this setting: instead of fixing the initial time to be 0, let it be a parameter t_0. We also treat x_0 as a parameter. Thinking discretely – for a numerical solution scheme – we may imagine that t_0 is restricted to take values $j\Delta t$, with $0 \le j \le N$ and $T = N\Delta t$. Similarly, x_0 is restricted to a lattice in R^n with mesh space Δx, and $\alpha(t)$ is defined only at times $j\Delta t$. Dynamic programming works backwards in time:

First Consider the problem with $t_0 = T$. In this case the dynamics is irrelevant. So are the control and the running cost. Whatever the value of x_0, the associated value of the cost is just $g(x_0)$.

Next Consider the problem with $t_0 = T - \Delta t$. Approximate the dynamics as

$$x(t + \Delta t) = x(t) + f(x(t), \alpha(t))\Delta t$$

For any fixed initial state $x_0 = x(t_0)$, the optimal control is now just a single vector (not a function), $\alpha(t_0)$. It is determined by minimizing the cost (of course we approximate the integral by a sum). Notice that the cost is easy to express as a function of $\alpha(t_0)$ since the discretized dynamics determines the state at time $t_0 + \Delta t = T$. Sometimes the cost can be minimized by hand using calculus; other times it can be found numerically by steepest descent (but beware of local minima!); other times a direct search of all possible values for $\alpha(t_0)$ is required. This calculation must be done for all possible values of x_0.

Next Consider the problem with $t_0 = T - 2\Delta t$. For any fixed initial state $x_0 = x(t_0)$, the optimal control is now represented by a pair of vectors $\alpha(t_0), \alpha(t_0 + \Delta t)$. However we can still determine it by solving a minimization problem involving just one vector $\alpha(t_0)$. Indeed, the cost is easy to express as a function of $\alpha(t_0)$: it equals the running cost from time t_0 to $t_0 + \Delta t$ [determined by calculating $x(t_0 + \Delta t)$ from the discretized dynamics] plus the optimal cost associated with using "initial state" $x(t_0 + \Delta t)$ and "initial time" $t_0 + \Delta t$ [this optimal cost was computed in the preceding paragraph]. This calculation determines the optimal value associated with any x_0, and also the optimal control that achieves it [once the optimal value of $\alpha(t_0)$ is determined, so is the value of $x(t_0 + \Delta t)$; the calculation in the preceding paragraph determined the associated optimal control $\alpha(t_0 + \Delta t)$].

Continue Work backward, one stage at a time. At the jth stage it is important to consider *all* possible "initial states" $x_0 = x(T - j\Delta t)$, or at least all those that might actually lie on an optimal trajectory. The information to be saved from the jth stage, when $t_0 = T - j\Delta t$, is the optimal cost and the associated optimal control $\alpha_0 = \alpha(T - j\Delta t)$ associated with each possible initial state $x_0 = x(T - j\Delta t)$.

The most awkward part of this algorithm is the need to consider *all* possible initial states x_0 at each stage: when x is vector valued this can be prohibitive (especially if Δx is small). The method is especially easy when the state is one-dimensional or the problem is intrinsically discrete rather than continuous. An example of the latter is a standard method for computing a shortest path between two nodes of a graph: Pick one of the nodes (call it an endpoint). Find all nodes that lie distance 1 from it, then all points that lie distance 2 from it, etc. Stop when the other endpoint appears in the set you come up with.

Students of math finance will have noticed by now that dynamic programming looks a lot like the binomial-tree method for valuing a European or American option. The resemblance is no coincidence. The biggest difference is that for the European option no optimization need be done at any point in the calculation; for the American option the optimization is simple – over just two alternatives, to exercise or not to exercise. This is due to the completeness of the underlying market model. In a multiperiod market that's not complete, there *is* an optimization to be done at each stage, namely an optimization over all risk-neutral probabilities. But we're getting ahead of ourselves – these stochastic control issues will be addressed later in the course. (Students not familiar with option pricing: don't worry, the concepts in this paragraph will be developed from scratch when we need them.)

When the state is vector-valued and continuous, the algorithm described above is basically a numerical scheme for solving a certain PDE. We shall shortly identify the PDE and discuss what its solutions can look like.

But first we digress to discuss some other types of optimal control problems. Since the initial state and time are now variables, we prefer to call them x and t rather than x_0 and t_0, and we represent the solution of ODE by a new name $y(s)$:

$$\dot{y}(s) = f(y(s), \alpha(s)) \text{ for } t < s < T \text{ with initial data } y(t) = x.$$

Sometimes we wish to emphasize the dependence of $y(s)$ on the initial value x, the initial time t, and the choice of control $\alpha(s), t < s < T$; in this case we write $y = y_{x,t,\alpha}(s)$. The control is typically restricted to take values in some specified set A, independent of s:

$$\alpha(s) \in A \text{ for all } s;$$

the set A must be specified along with the dynamics f. Sometimes it is natural to impose *state constraints*, i.e. to require that the state $y(s)$ stay in some specified set Y:

$$y_{x,t,\alpha}(s) \in Y \text{ for all } s;$$

when present, this requirement restricts the set of admissible controls $\alpha(s)$. The value of the minimum cost, viewed as a function of the initial state and time, is called the **value function**. The problem discussed above is sometimes called the **finite horizon** problem; its value function is

$$u(x,t) = \min_{\alpha} \left\{ \int_t^T h(y_{x,t,\alpha}(s), \alpha(s))\, ds + g(y_{x,t,\alpha}(T)) \right\}.$$

For the analogous **infinite horizon** problem we may fix the starting time to be 0, so the value function depends only on the spatial variable x:

$$u(x) = \min_{\alpha} \int_0^\infty h(y_{x,0,\alpha}(s), \alpha(s)) e^{-s}\, ds$$

(one can, of course, introduce a discount factor e^{t-s} even in a finite horizon problem). The **minimum time** problem minimizes the time it takes $y(s)$ to travel from x to some target set $\mathcal{T}$: its value function is

$$u(x) = \min_\alpha \{\text{time at which } y_{x,0,\alpha}(s) \text{ first arrives in } \mathcal{T}\}.$$

The minimum time problem is somewhat singular: if, for some x, the solution starting at x cannot arrive in $\mathcal{T}$ (no matter what the control) then the value is undefined. The **discounted minimum time** problem avoids this problem: its value function is

$$u(x) = \min_\alpha \int_0^{\tau(x,\alpha)} e^{-s}\, ds$$

where $\tau(x,\alpha)$ is the time that $y_{x,0,\alpha}(s)$ first arrives in $\mathcal{T}$, or infinity if it never arrives. Notice that the integral can be evaluated: the quantity being minimized is $\int_0^{\tau(x,\alpha)} e^{-s}\, ds = 1 - e^{-\tau(x,\alpha)}$. So we're still minimizing the arrival time, but the value function is $1 - \exp(-\text{arrival time})$ instead of the arrival time itself.

Another digression: let's give an explicit example of a control problem from finance. The following example is closely connected with Merton's theory of optimal investment and consumption, which we'll be discussing later. Differences: our example is simpler than Merton's because its dynamics is deterministic rather than stochastic; on the other hand it includes transaction costs and a solvency constraint, subtleties that have only recently been addressed in Merton's context. Our example is a stripped-down version of the problem discussed in S. Shreve, H. Soner, and G. Xu, "Optimal investment and consumption with two bonds and transaction costs," *Mathematical Finance* Vol. 1, No. 3, 1991, 53-84.

Suppose an investor can choose between two different investment opportunities:

- a money-market account, paying constant interest r, and
- a high-yield account, paying constant interest $R > r$.

The investor can move money between the two accounts, but in doing so he incurs a transaction fee proportional to the amount of money moved:

- when moving money from money-market to high-yield, X dollars in money-market becomes $(1-\mu)X$ dollars in high-yield (μX is the transaction cost);
- when moving money from high-yield to money-market, Y dollars in high-yield becomes $(1-\mu)Y$ dollars in money-market (μY is the transaction cost).

The investor can remove money from the investment fund only by taking it out of the money market account; there is no transaction fee associated with such consumption. The investor can take short positions in either account, however (to avoid the obvious arbitrage) we impose a "solvency constraint:" liquidation into money market should not leave him in debt. When X is the money-market position and Y is the high-yield position, the solvency condition says:

- if $Y \geq 0$ then $X + (1-\mu)Y \geq 0$. (Liquidation involves turning Y dollars in high-yield into $(1-\mu)Y$ dollars in money-market; this must be sufficient to pay off any money-market debt).
- if $Y \leq 0$ then $X + Y/(1-\mu) \geq 0$. (Liquidation involves paying off the high-yield debt by removing $|Y|/(1-\mu)$ dollars from money-market; the resulting money-market balance must not be negative.)

The investor's goal is to maximize the discounted utility of his total future consumption. Let us formulate this as a control problem. The *state* is an R^2-valued function of time, $(X(t), Y(t))$, where

- $X(t)$ = money-market position at time t,
- $Y(t)$ = high-yield position at time t;

the solvency condition is a *state constraint.* The *control* is an R^3-valued function of time, $(\alpha(t), \beta(t), \gamma(t))$, where

- $\alpha(t) \geq 0$ is the rate at which money is being moved from money-market to high-yield at time t,
- $\beta(t) \geq 0$ is the rate at which money is being moved from high-yield to money-market at time t,
- $\gamma(t) \geq 0$ is the consumption rate at time t.

The evolution equation is

$$\begin{aligned} dX/dt &= rX - \alpha + (1-\mu)\beta - \gamma \\ dY/dt &= RY + (1-\mu)\alpha - \beta, \end{aligned}$$

with initial conditions $X(0) = X_0, Y(0) = Y_0$. Using a power-law utility and discount rate 1, the investor's goal of maximizing lifetime utility becomes

$$\max_{\alpha,\beta,\gamma} \int_0^\infty e^{-s} \gamma^p(s)\, ds.$$

This is an *infinite-horizon* problem; its value function is $u(X_0, Y_0)$ = maximum lifetime utility as a function of initial position.

Some remarks: (1) We permit the investor to "move" money from money-market to high-yield even when $X < 0$; this amounts to borrowing money at the money-market rate to purchase the high-yield investment. Similarly he can "move" money from high-yield to money-market even when $Y < 0$. (2) If the investor moves a certain amount of money all at once at time t_* then α or β is formally infinite (like a "delta-function") at t_* and X, Y are discontinuous; such policies can be approximated, of course, by continuous ones. (3) The exponent p in our expression for lifetime utility should satisfy $0 < p < 1$, since the benefit $\gamma^p\, dt$ associated with consuming $\gamma\, dt$ in a time interval dt should be an increasing, concave function of γ (the second derivative is negative: law of diminishing marginal return from increased consumption). We will return to this problem – describing its solution – a little later.

Enough digression. Let's return to the mathematical question raised earlier: what PDE is solved by the dynamic programming algorithm presented above, in the continuous-time, continuous-space limit? We have discussed a number of different problems (minimize cost, maximize utility, finite vs infinite horizon, minimum time, etc.) and essentially the same method can be applied to all of them. Let's concentrate on the version that maximizes utility with a finite horizon:

$$\text{maximize} \left\{ \int_t^T h(y(s), \alpha(s))\, ds + g(y(T)) \right\}$$

over controls restricted only by $\alpha(t) \in A$, where

$$dy/ds = f(y(s), \alpha(s)) \text{ for } t < s < T \text{ and } y(t) = x.$$

Its value function $u(x, t)$ gives the maximal utility as a function of the initial time and state. The answer to our question is this: u solves the *Hamilton-Jacobi-Bellman* equation

$$u_t + H(\nabla u, x) = 0 \quad \text{for } t < T$$

with

$$u(x, T) = g(x) \quad \text{at } t = T,$$

where H is defined by

$$H(p, x) = \max_{a \in A} \{ f(x, a) \cdot p + h(x, a) \}.$$

(Note that p is a vector with the same dimensionality as x; a is a vector with the same dimensionality as α.)

To explain, we start with the *dynamic programming principle*, which captures the essential idea of our discrete scheme. It says:

$$u(x, t) = \max_{\alpha} \left\{ \int_t^{t'} h(y_{x,t,\alpha}(s), \alpha(s))\, ds + u(y_{x,t,\alpha}(t'), t') \right\} \tag{1}$$

whenever $t < t' < T$. The justification is easy, especially if we assume that an optimal control exists (this case captures the main idea; see Evans for a more careful proof, without this hypothesis). Suppose the optimal cost starting at x at time t is achieved by an optimal control $\alpha_{x,t}(s)$. Then the restriction of this control to any subinterval $t' < s < T$ must be optimal for its starting time t' and starting position $y_{x,t,\alpha}(t')$. Indeed, if it weren't then there would be a new control $\alpha'(s)$ which agreed with α for $t < s < t'$ but did better for $t' < s < T$. Since the cost is additive – the running cost is $\int_t^T h(y, \alpha)\, ds = \int_t^{t'} h(y, \alpha)\, ds + \int_{t'}^T h(y, \alpha)\, ds$ – this new control would be better for the entire time period, contradicting the optimality of α. Therefore in defining $u(x, t)$ as the optimal cost, we can restrict our attention to controls that are optimal from time t' on. This leads immediately to (1).

Now a heuristic justification of the Hamilton-Jacobi-Bellman equation. The basic idea is to apply the dynamic programming principle with $t' = t + \Delta t$ and let $\Delta t \to 0$. Our argument is heuristic because we assume (i) u is differentiable, and (ii) the optimal control is adequately approximated by one which is constant for $t < s < t+\Delta t$. (Our goal, as usual, is to capture the central idea, referring to Evans for a more rigorous treatment.) Since Δt is small, the integral on the right hand side of (1) can be approximated by $h(x,a)\Delta t$, where $a \in A$ is the (constant) value of α for $t < s < t+\Delta t$. Using a similar approximation for the dynamics, the dynamic programming principle gives

$$u(x,t) \geq h(x,a)\Delta t + u(x + f(x,a)\Delta t, t+\Delta t) + \text{errors we wish to ignore}$$

with equality when a is chosen optimally. Using the first-order Taylor expansion of u this becomes

$$u(x,t) \geq h(x,a)\Delta t + u(x,t) + (\nabla u \cdot f(x,a) + u_t)\Delta t + \text{error terms}$$

with equality when a is optimal. In the limit $\Delta t \to 0$ this gives

$$0 = u_t + \max_{a\in A}\{\nabla u \cdot f(x,a) + h(x,a)\},$$

i.e. $u_t + H(\nabla u, x) = 0$ with H as asserted above. The final-time condition is obvious: if $t = T$ then the dynamics is irrelevant, and the total utility is just $g(x)$.

The equation simplifies to $u_t + H(\nabla u) = 0$ if f and h are both independent of x. Let's focus on this case for a while. The Hamiltonian H is necessarily convex in ∇u, since the formula

$$H(p) = \max_{a\in A}\{f(a)\cdot p + h(a)\}$$

expresses it as a maximum of linear functions of p. The function H does not determine f and h – different f's and h's can lead to the same H. But for any convex H there's an especially simple choice of an associated f and h, namely

$$f_*(a) = a, \quad h_*(a) = \min_p\{H(p) - a\cdot p\}.$$

Explanation of the latter: once we fix $f(a) = a$, the formula for H says $H(p) \geq a\cdot p + h(a)$ for all a and p, with equality when $a = a(p)$ is optimal. Rewrite this as $h(a) \leq H(p) - a\cdot p$. Our h_* is the the largest possible value for $h(a)$, obtained by minimizing $H(p) - a\cdot p$ over p. This calculation is closely related to "convex duality" and the "Fenchel transform." Notice that h_* is concave, as a utility should be, since it is a minimum of linear functions.

The optimal control problem associated with $f = f_*$ and $h = h_*$ is easy to solve. Indeed, we claim that whenever $f(a) = a$ and $h(a)$ is concave, the optimal control is constant and the associated trajectory is a constant-velocity path. Indeed, $f(a) = a$ means a is the *velocity* of the path $y(s)$. The concavity of h gives

$$h[\text{average velocity}] \geq \text{average of } h[\text{velocity}].$$

Notice moreover that the average velocity of a path depends only on its endpoints, since

$$\frac{1}{T-t}\int_t^T \frac{dy}{ds}\,ds = \frac{1}{T-t}(y(T) - y(t)).$$

Thus replacing any path by one with the same endpoints and constant velocity can only improve the utility. We thus arrive at the *Hopf-Lax* solution formula: when $f(a) = a$ and $h(a)$ is concave,

$$u(x,t) = \max_z \left\{ (T-t)h\left(\frac{z-x}{T-t}\right) + g(z) \right\}.$$

Here z represents the state at time T – the only remaining unknown – and $(z-x)/(T-t)$ is the velocity of the associated path starting at x at time t and ending at z at time T.

Let's bring this down to earth by considering a very specific example: $f(a) = a$, $H(p) = \frac{1}{2}|p|^2$, and $h(p) = -\frac{1}{2}|a|^2$. Then the Hamilton-Jacobi-Bellman equation is

$$u_t + \frac{1}{2}|\nabla u|^2 = 0, \quad u(x,T) = g(x)$$

and the solution formula is

$$u(x,t) = \max_z \left\{ g(z) - \frac{|z-x|^2}{2(T-t)} \right\}.$$

An important fact is immediately evident: the Hamilton-Jacobi-Bellman equation has many (almost-everywhere) solutions, only one of which agrees with the solution formula. For example, suppose $g = 0$. Then the solution formula gives $u(x,t) = 0$, which does solve the Hamilton-Jacobi equation. However the PDE has lots of other solutions: for example the function

$$u(x,t) = \begin{cases} \frac{1}{2}(T-t) - |x| & \text{if } |x| \leq \frac{1}{2}(T-t) \\ 0 & \text{otherwise.} \end{cases}$$

This example is easy to generalize, yielding infinitely many "solutions" of the PDE, all equal to 0 at $t = T$.

One might think, at first, that the issue is regularity. Might the correct solution be a unique smooth solution? The answer is no: the correct solution can easily fail to be smooth. An example: when $g(y) = |y|$, the solution formula gives

$$u(x,t) = \frac{T-t}{2} + |x|$$

(the optimal z is $z = x + (T-t)x/|x|$). The lack of smoothness arises through non-smooth dependence of the optimal z on x.

Another point to note: the Hamilton-Jacobi equation is *nonlinear*. If u_1 solves it with final data g_1 and u_2 solves it with final data g_2, we should not expect $u_1 + u_2$ to solve it with final data $g_1 + g_2$. When $H(p) = H(-p)$, for example when $H(p) = |p|^2/2$, one might imagine that if u is the correct solution with final data g then $-u$ is the correct solution with final data $-g$. But even this is false: when $g(y) = -|y|$ the solution formula for $u_t + \frac{1}{2}|\nabla u|^2 = 0$ gives

$$u(x,t) = \begin{cases} (T-t)/2 - |x| & \text{if } |x| \geq (T-t) \\ -|x|^2/2(T-t) & \text{otherwise} \end{cases}$$

(the optimal z is $z = x - (T-t)x/|x|$ if $|x| \geq (T-t)$, $z = 0$ otherwise).

What use, then, is the Hamilton-Jacobi equation? Its uses are basically three-fold:

(a) The value function must satisfy it, wherever the value function is smooth.

(b) A smooth solution of the Hamilton-Jacobi equation (or even an associated inequality) can be used to demonstrate the optimality of a proposed optimal control.

(c) There is a more sophisticated notion of "solution" of a Hamilton-Jacobi equation, namely the *viscosity solution*. Viscosity solutions exist, are unique, and when H is convex they agree with the dynamic programming solution discussed above.

We've basically discussed (a). Point (c) is an interesting story, which we'll discuss in Section 2. To explain (b) – whose applications are sometimes known as *verification theorems* – let's consider an easily-visualized special case of the minimum time problem: starting at a point x in R^n, travel with speed ≤ 1 so as to arrive in a set $\mathcal{T}$ as quickly as possible. The optimal strategy, of course, is to travel with constant velocity toward the point in $\mathcal{T}$ that is closest to x, and the value function (the arrival time) is

$$u(x,t) = \begin{cases} 0 & \text{for } x \in \mathcal{T} \\ \text{dist}(x, \mathcal{T}) & \text{for } x \notin \mathcal{T}. \end{cases}$$

Notice that $\text{dist}(x, \mathcal{T})$ can easily fail to be smooth, even when the shape of $\mathcal{T}$ is very smooth. We thus have another example of how nonsmoothness arises naturally in the solutions to dynamic programming problems. This example fits right into the dynamic programming framework of course: the state evolves by

$$dy/ds = \alpha(s), \quad y(t) = x,$$

and the controls are restricted by

$$\alpha(s) \in A \quad \text{for all } s$$

with A being the unit ball. The associated Hamilton-Jacobi equation is $1 - |\nabla u| = 0$. We can see this – and generalize it – by considering the minimum arrival time problem for a more general state equation

$$dy/ds = f(y, \alpha), \quad y(0) = x.$$

Arguing much as we did for the other problem, we see that the value function (the time it takes to arrive at $\mathcal{T}$) should satisfy

$$u(x) \leq \Delta t + u(x + f(x,a)\Delta t) + \text{error terms}$$

for any $a \in A$, with equality when a is optimal. Using Taylor expansion this becomes

$$u(x) \leq \Delta t + u(x) + \nabla u \cdot f(x,a)\Delta t + \text{error terms}.$$

Optimizing over a and letting $\Delta t \to 0$ we get

$$1 + \min_{a \in A}\{f(x,a) \cdot \nabla u\} = 0.$$

This can be written as $1 - H(x, \nabla u) = 0$ with

$$H(x,p) = \max_{a \in A}\{-f(x,a) \cdot p\}.$$

When $f(x,a) = a$ and A is the unit ball we get $1 - |\nabla u| = 0$, as expected.

Now for a simple example of a verification theorem: suppose we believe a control $\alpha_*(s)$ is optimal for x. To prove we are right, it suffices to find a smooth function v such that

(a) $v \leq 0$ on $\partial\mathcal{T}$,

(b) $H(y, \nabla v(y)) \leq 1$ for all $y \notin \mathcal{T}$, and

(c) $-f(y_*(s), \alpha_*(s)) \cdot \nabla v(y_*(s)) = 1$ along the path y_* associated with control α_*, and $v = 0$ where this path first reaches $\mathcal{T}$.

For then consider an arbitrary control $\alpha(s)$ and its associated path $y(s)$. This path starts at x when $s = 0$; suppose it arrives at $\mathcal{T}$ when $s = S$. We have

$$-\frac{d}{ds}v(y(s)) = -f(y(s), \alpha(s)) \cdot \nabla v(y(s)) \leq H(\nabla v(y(s))) \leq 1$$

using (b). Integrating this from $s = 0$ to $s = S$ gives

$$S \geq v(x) - v(y(S)) \geq v(x),$$

using (a). When we repeat this calculation for the special path y_* associated with α_* the inequalities become equalities, by (c), so the arrival time S_* satisfies

$$S_* = v(x)$$

Thus the path is indeed optimal, and $S_* = u(x)$ is the minimal arrival time.

In the special case of our geometrical example $1 - |\nabla u| = 0$ the natural choice of v is linear, with unit-length gradient parallel to $x - z$ where z is the point of $\mathcal{T}$ closest to x. It is instructive to repeat the verification argument directly for this example, using this v.

Similar verification principles exist for all the dynamic programming problems considered above.