Electronic Journal of Probability

Vol. 6 (2001) Paper no. 20, pages 1–21.

Journal URL
http://www.math.washington.edu/~ejpecp/
Paper URL
http://www.math.washington.edu/~ejpecp/EjpVol6/paper20.abs.html

WEAK SOLUTIONS FOR A SIMPLE HYPERBOLIC SYSTEM

Owen D. Lyne
School of Mathematical Sciences, University of Nottingham
University Park, Nottingham NG7 2RD, United Kingdom
owen.lyne@nottingham.ac.uk

David Williams
Brookside, Reynoldston, Swansea SA3 1AD, United Kingdom
dswilliams@markov.fsnet.co.uk

Abstract The model studied concerns a simple first-order *hyperbolic* system. The solutions in which one is most interested have discontinuities which persist for all time, and therefore need to be interpreted as *weak* solutions. We demonstrate existence and uniqueness for such weak solutions, identifying a canonical '*exact*' solution which is *everywhere* defined. The direct method used is guided by the theory of measure-valued diffusions. The method is more effective than the method of characteristics, and has the advantage that it leads immediately to the McKean representation without recourse to Itô's formula.

We then conduct computer studies of our model, both by integration schemes (which *do* use characteristics) and by 'random simulation'.

Keywords Weak solutions. Travelling waves. Martingales. Branching processses

AMS subject classification 35L45; 60J27; 60G44

Research conducted while both authors were at the University of Bath.

Submitted to EJP on January 22, 2001. Final version accepted on August 15, 2001.

1 Introduction

After a change of the original notation with u replacing $1-u$, the FKPP equation (Fisher [6], Kolmogorov, Petrovskii & Piskunov [9]) reads:

$$\frac{\partial u}{\partial t} = \tfrac{1}{2}\frac{\partial^2 u}{\partial x^2} + u^2 - u, \qquad u(0,x) = f(x).$$

McKean [12, 13] showed that one can prove existence and uniqueness results for certain $[0,1]$-valued solutions by using martingales to describe u as an explicit functional of a certain branching Brownian motion. He was thereby able to obtain results on convergence to travelling waves for suitable initial data f. Neveu [15] gives an important completion of McKean's treatment.

Two recent papers — Champneys, Harris, Toland, Warren & Williams [2] and Lyne [10] — have each presented both analytic and probabilistic studies of simple extensions of the FKPP equation. These papers used the probabilist's golden rule that Itô's formula leads to martingales (see, for example, Rogers & Williams [17]). But the uses of Itô's formula involved the 'formal generator' of the branching process in a way which might cause some unease to analysts. The model studied by Lyne concerns a simple first-order *hyperbolic* system (of similar nature to those studied in Dunbar [4], Othmer, Dunbar & Alt [16], Holmes [8] and Hadeler [7] — see the discussion in Lyne [11] for the relations between the models), a generalization of which we consider here. The solutions in which one is most interested have discontinuities which persist for all time, and therefore need to be interpreted as *weak* solutions. Section 2 settles existence and uniqueness for such weak solutions, identifying a canonical '*exact*' solution which is *everywhere* defined. The direct method used is guided by the theory of measure-valued diffusions, MVDs, (see, for example, Dawson [3] and Dynkin [5]). (We stress that no knowledge of MVD theory is assumed here.) The method is more effective than the method of characteristics, and has the advantage that it leads immediately to the McKean representation without recourse to Itô's formula.

Having (we hope) satisfied the requirements of analysis in Section 2, we turn in Section 3 — with more freedom — to the question of computer studies of our model, both by integration schemes (which *do* use characteristics) and by 'random simulation'. Numerical analysts would wish for more rigour in Section 3. We settle for a certain amount of cross-checking.

Notational point. We *never* use u_t to denote $\frac{\partial u}{\partial t}$, rather u_t denotes u at time t.

2 Theoretical results

2.1 Our hyperbolic system

Let I be a finite set (with the discrete topology); and let B and R be functions of I with $R \geq 0$. Let Q be an $I \times I$ matrix with non-negative off-diagonal elements and zero row sums.

Let f be a Borel function on $\mathbb{R} \times I$ with $0 \leq f \leq 1$. Let u, written $(t,x,j) \mapsto u_t(x,j)$ and regarded as a column vector in j when multiplied by Q, be a Borel function on $[0,\infty) \times \mathbb{R} \times I$ with $0 \leq u \leq 1$. Suppose that u is a weak solution of

$$\frac{\partial u}{\partial t} = B\frac{\partial u}{\partial x} + Qu + R(u^2 - u), \quad \text{“}u_0(x,j) = f(x,j)\text{”}. \tag{1}$$

In full, the first equation reads:

$$\frac{\partial}{\partial t}u_t(x,j) = B(j)\frac{\partial}{\partial x}u_t(x,j) + \sum_k Q(j,k)u_t(x,k) + R(j)\left[u_t(x,j)^2 - u_t(x,j)\right].$$

By the statement that u is a weak solution, we mean that for $t>0$ and a test function $\varphi \in C_K^{1,1,0}([0,t]\times\mathbb{R}\times I)$ (that is a function of compact support, continuously differentiable in $[0,t]$ and in space) thought of as a row vector in j,

$$\int_0^t \int_x \sum_j \left\{\frac{\partial\varphi}{\partial s} - \frac{\partial\varphi}{\partial x}B + \varphi(Q-R) + \varphi R u\right\}_s (x,j)u_s(x,j)dxdt \tag{2}$$

$$= \int_x \sum_j \varphi_t(x,j)u_t(x,j)dx - \int_x \sum_j \varphi_0(x,j)f(x,j)dx.$$

2.2 Analytic statement of some results

Shortly we shall shortly reformulate these results probabilistically.

Introduce the unique one-parameter (Markov) semigroup $\{P_t^{-R} : t \geq 0\}$ acting on $C_b(\mathbb{R}\times I)$ (suffix b standing for 'bounded') such that if $h(x,j) = e^{i\theta x}g(j)$, where $\theta\in\mathbb{R}$ and g is a function (or column vector) on I, then

$$\left(P_t^{-R}h\right)(x,j) = e^{i\theta x}\left(\exp\{(i\theta B + Q - R)t\}g\right)(j). \tag{3}$$

In regard to the existence of $\{P_t^{-R} : t\geq 0\}$, see the discussion around equation (5) below. By the Riesz representation theorem, $\{P_t^{-R} : t \geq 0\}$ has a canonical extension to a semigroup on $\mathcal{B}_b(\mathbb{R}\times I)$, the space of bounded Borel functions on $\mathbb{R}\times I$.

Equation (2) implies that for each $t>0$, u satisfies:

$$u_t = P_t^{-R}f + \int_0^t P_{t-s}^{-R}(Ru_s^2)ds, \tag{4}$$

for almost every x. This is proved as follows. Standard Fourier theory shows that for $\psi_0(\cdot,\cdot)$ in $C_K^\infty(\mathbb{R}\times I)$,

$$\psi_r(x,j)dx := \int dx_0 \sum_{j_0} \psi_0(x_0,j_0)P_r^{-R}(x_0,j_0;dx,j) \quad (0\leq r\leq t)$$

defines $\psi_\cdot(\cdot,\cdot)\in C_K^{1,1,0}([0,t]\times\mathbb{R}\times I)$ with

$$\frac{\partial\psi}{\partial r} = -\frac{\partial\psi}{\partial x}B + \psi(Q-R).$$

Now take $\varphi_s(x,j) = \psi_{t-s}(x,j)$ in equation (2).

But now define

$$\begin{aligned} u_t^{(1)} &:= P_t^{-R}f, \\ u_t^{(n+1)} &:= P_t^{-R}f + \int_0^t P_s^{-R}\left(R {u_{t-s}^{(n)}}^2\right)ds, \quad (n\geq 1). \end{aligned}$$

Then it is almost immediate by the usual Picard/Gronwall argument that $u^* := \lim u^{(n)}$ exists *monotonically and uniformly on each* $[0,t]\times\mathbb{R}\times I$, and so gives *the exact* solution of equation (4): it is a solution in which there are no 'exceptional sets'; and if v is another exact solution to equation (4) with $0 \le v \le 1$, then v is equal to u^* *everywhere*. Moreover, u^* is a weak solution of equation (1).

2.3 Probabilistic interpretations/proofs

Let $\{\eta_t : t \ge 0\}$ be a Markov chain on I with Q-matrix Q. Define

$$\xi_t := \xi_0 + \int_0^t B(\eta_s)ds,$$

and set

$$\left(P_t^{-R}h\right)(x,j) := \mathbb{E}^{x,j}\, h(\xi_t,\eta_t) \exp\left\{-\int_0^t R(\eta_s)ds\right\},$$

where $\mathbb{E}^{x,j}$ is the measure corresponding to starting position $(\xi_0,\eta_0) = (x,j)$. Let us check that this agrees with the semigroup of equation (3). Fix $\theta \in \mathbb{R}$. For a vector g on I, define

$$Z_t(g) := \exp\left(-\int_0^t R(\eta_s)ds\right) e^{i\theta\xi_t} g(\eta)_t. \tag{5}$$

If we write $\doteq$ to signify equality modulo differentials of local martingales, then (see Rogers & Williams [17])

$$d\{g(\eta_t)\} \doteq (Qg)(\eta_t)dt,$$

and

$$dZ_t(g) \doteq \exp\left(-\int_0^t R(\eta_s)ds\right) e^{i\theta\xi_t}\left([-R+i\theta B+Q]g\right)(\eta_t)dt. \tag{6}$$

In fact, the difference between the integrals of the two sides of equation (6) is bounded on each $[0,t]$, and so is not just a local, but a true, martingale. Hence

$$\frac{d}{dt}(S_t g) = S_t\,(i\theta B + Q - R)g,$$

where S_t is the linear map on vectors on I defined by

$$S_t g := \mathbb{E}^{0,j} Z_t(g).$$

Since S_0 is the identity, equation (3) now follows.

Guided by McKean, we now construct a branching Markov process related to the hyperbolic equation (1). Time 0 sees the birth of one particle, labelled 1, which has 'type' $Y_1(0)$ in I and 'position' $X_1(0)$ in $\mathbb{R}$. At time $t \ge 0$, there are $N(t)$ particles which, when labelled in order of birth, have 'types' $Y_1(t),\dots,Y_{N(t)}(t)$ in I, and positions $X_1(t),\dots,X_{N(t)}(t)$ in $\mathbb{R}$. The type of each particle behaves (independently of previous history, of the behaviour of other particles currently alive, etc) as a Markov chain on I with Q-matrix Q. A particle of type j moves on $\mathbb{R}$ with constant speed $B(j)$, and gives birth to a new particle of its own type with rate $R(j)$, so that in small time h, independently of 'everything else', it gives birth with probability

$R(j)h + o(h)$. Particles live forever, once born. We write $\mathbb{P}^{x,j}$ and $\mathbb{E}^{x,j}$ for the probability and expectation corresponding to the situation when $X_1(0) = x$ and $Y_1(0) = j$.

With f as our 'initial value for u_0', let

$$\Pi(t) := \prod_1^{N(t)} f\Big(X_k(t), Y_k(t)\Big), \quad v_t(x,j) := \mathbb{E}^{x,j}\,\Pi(t).$$

We now utilise an obvious argument. Let T be the time of the first birth after time 0, so that T is the birth-time of particle 2. Because of the rôle of R as birth-rate function, we have

$$\mathbb{P}^{x,j}\left(T \in ds \,\Big|\, Y_1(r) : r \le s\right) = R(Y_1(s)) \exp\left\{-\int_0^t R(Y_1(r))dr\right\} ds.$$

From time T on, the family tree of particle 2 evolves independently of its complement in the family tree of particle 1. We therefore have

$$\mathbb{E}^{x,j}\left(\Pi(t) \,\Big|\, T = s;\, Y_1(r) : r \le s\right) = v_{t-s}(X_1(s), Y_1(s))^2 \quad (s \le t).$$

Hence, for $s \le t$,

$$\mathbb{E}^{x,j}\left(\Pi(t); T \in ds \,\Big|\, Y_1(r) : r \le s\right)$$
$$= R(Y_1(s)) v_{t-s}(X_1(s), Y_1(s))^2 \exp\left\{-\int_0^t R(Y_1(r))dr\right\} ds,$$

so that

$$\begin{aligned}
\mathbb{E}^{x,j}\left(\Pi(t); T \in ds\right) &= \mathbb{E}^{x,j} R(\eta_s) v_{t-s}(\xi_s, \eta_s)^2 \exp\left\{-\int_0^t R(\eta_r)dr\right\} ds \\
&= \left\{P_s^{-R}\left(R v_{t-s}^2\right)\right\}(x,j) ds.
\end{aligned}$$

Since

$$\mathbb{E}^{x,j}\left(\Pi(t); T > t\right) = \left(P_t^{-R} F\right)(x,j),$$

we see that v satisfies equation (4) exactly. Hence $v = u^*$ *everywhere*, and we have the McKean representation

$$u_t^*(x,j) = \mathbb{E}^{x,j}\,\Pi(t).$$

2.4 Convergence to travelling waves: the easy case

Suppose that

$$f(x,j) = \begin{cases} 1 & \text{if } x > 0, \\ 0 & \text{if } x \le 0, \end{cases}$$

so that

$$u_t(x,j) = \mathbb{P}^{x,j}(L(t) > 0) = \mathbb{P}^{0,j}(L(t) + x > 0)$$

where

$$L(t) := \min\{X_k(t) : k \le N(t)\}.$$

Condition (†)

We consider the very special situation in which

$$Q \text{ is irreducible and there is a state } j_0 \in I \text{ with}$$

$$B(j_0) < B(j) \text{ for } j \neq j_0 \text{ and for which } R(j_0) > -Q(j_0, j_0).$$

We will refer to these conditions as (†).

When (†) holds there will almost surely exist at some time a particle of type j_0 which has an infinite '*line of descent*' consisting entirely of particles of type j_0. Thus there will be a *random* interval $[\sigma, \infty)$, which we choose to be maximal, such that for some random constant A,

$$L(t) - B(j_0)t = A \quad \text{for} \quad t \in [\sigma, \infty).$$

Then

$$\begin{aligned} u_t(x - B(j_0)t, j) &= \mathbb{P}^{0,j}(x + L(t) - B(j_0)t > 0) \\ \to w(x, j) &= \mathbb{P}^{0,j}(A > -x), \end{aligned}$$

and $w(x + tB(j_0), j)$ is a travelling-wave solution of equation (1).
In this case,

$$u_t(x, j_0) = 1 \quad \text{if} \quad x > -tB(j_0),$$
$$u_t(x, j_0) = 1 - \mathbb{P}^{0,j_0}(L(t) = tB(j_0)) < 1 \quad \text{if} \quad x = -tB(j_0),$$

and the jump $\mathbb{P}^{0,j_0}(L(t) = tB(j_0))$ at the '*characteristic point*' $x = -tB(j_0)$ converges as $t \to \infty$ to

$$\mathbb{P}^{0,j_0}(\sigma = 0) = \frac{R(j_0) + Q(j_0, j_0)}{R(j_0)}.$$

For numerical and simulation studies of such a case, see section 3 below.

2.5 The difficult cases

It is hoped to make the difficult cases when (†) fails to hold the subject of another paper giving direct proofs for this simple situation of results

$$u_{t+ct-a(t)}(x, j) \to w(x, j) \quad \text{where} \quad a(t) = o(t).$$

Indeed, $a(t)$ may behave like a multiple of $\log t$ or of $\log \log t$. Such results follow from deep results in existing literature. The classic paper on the '*logarithmic correction*' for the Fisher equation is that of Bramson [1].

3 Numerical analysis

We use both probabilistic simulation of the branching process and finite difference methods (the most effective being upwinding along characteristics) to study the initial value problem for the case where $I = \{1, 2\}$, that is, we have a pair of coupled first order PDEs (as studied in Lyne [10]). As in Dunbar [4] and Holmes [8], these can be rewritten as a single second order PDE which is a generalization of the telegraph equation. It is convenient to change to moving coordinates (moving at a speed of $\frac{1}{2}(B(1)+B(2))$) and then re-scale time so that the coefficients of $\frac{\partial u}{\partial x}$ are 1 and -1. This is possible unless $B(1) = B(2)$ — this case reduces to a pair of first order ODEs and is dealt with by Lyne [10]. For the remainder of this paper, we shall denote $u_t(x, 1)$ by $u(t, x)$ and $u_t(x, 2)$ by $v(t, x)$ and set $B(1) = 1, B(2) = -1$. Particularly interesting is Heaviside initial data, that is,

$$u(0, x) = v(0, x) = \begin{cases} 1 & \text{if } x > 0, \\ 0 & \text{if } x \leq 0. \end{cases}$$

We present first the skeleton of the `C` program used to produce the numerical solution plotted in Figures 1 and 4. It implements a naive Euler method along the characteristics of the system, and a modification of the Euler method. The figures were produced using the modified method, but output of the two methods is practically indistinguishable.

```
/*  EULER METHODS
This is only part of a program.

Use of naive Euler methods for a simple hyperbolic system

   du/dt =  du/dx + f(u,v),  f(u,v) = q1 (v-u) + r1 u(u-1);
   dv/dt = -dv/dx + g(u,v),  g(u,v) = q2 (u-v) + r2 v(v-1);

with Heaviside initial data
     u(0,x) = v(0,x) =  1 for x >  0,
                        0 for x <= 0.

After the nth step, u[k] = uu[k+500] (‘idioC’!) represents
     u(nh, 2kh)     if n is even;
     u(nh, (2k-1)h) if n is odd.

So, as it were, u[n,k] (that is, u[k] after n time steps)
corresponds to the pattern

t=2h            [2,-1]          [2,0]         [2,1]
t= h                    [1,0]          [1,1]
t= 0            [0,-1]          [0,0]         [0,1]

x-value          -2h      -h      0      h     2h
```

```
This is suited to integrating along the characteristics,
but we have to watch the parity.

nu denotes the next u-array, that is, u one time step later.

We consider the values -480 <= k <= 480; 0 <= n <= 960 (= bign),
with printouts every 160 (=gap) steps.                                     */

#define bign 960
#define gap 160
double q1, q2, r1, r2, t;
int a, b, n;  /* a and b are x-values where there are discontinuities */
double uu[1001], vv[1001], nuu[1001], nvv[1001];
double  *u= &uu[500]; double  *v= &vv[500];
double *nu=&nuu[500]; double *nv=&nvv[500];

int k, j, parity, method; double h;

void Solve(void); /*  calls up Display when appropriate */
void Euler1(void); void Euler2(void); double Trim(double z);

void Solve(void)
{  double temp;

   /* Initialize  */
   for(k=-485; k<= 0; k++) {u[k] = 0.0; v[k] = 0.0;}
   for(k=1; k<= 485; k++)  {u[k] = 1.0; v[k] = 1.0;}

   parity=0;
   for(n=1; n<=bign; n++)
   {  a = 1 - (n/2); b=(n+1)/2; parity = 1 - parity;

      for(k=a; k<= b; k++)
      {  j = k + 1 - parity;
         (method == 1)? Euler1(): Euler2();  }

      /* Stabilize */
      for (k=a; k<=b; k++)
      {  u[k] = Trim(nu[k]); v[k] = Trim(nv[k]);  }

      /* Display */
      if (n % gap == 0) {t = n * h; Display();}
   }
}

void Euler1(void) /* the most naive updating possible */
```

```
{  nu[k] = u[j] + h * f(u[j],v[j]);
   nv[k] = v[j-1] + h * g(u[j-1], v[j-1]);
}

void Euler2(void)  /* a refinement of the method */
{  double nu_temp, nv_temp;
   nu_temp = u[j] + h * f(u[j],v[j]);
   nv_temp = v[j-1] + h * g(u[j-1], v[j-1]);
   nu[k] = u[j] + 0.5*h*(f(u[j],v[j]) + f(nu_temp, nv_temp));
   nv[k] = v[j-1]+0.5*h*(g(u[j-1], v[j-1])+g(nu_temp, nv_temp));
}

double Trim(double z)
{  if (z>1.0) return 1.0
   else if (z<0.0) return 0.0
   else return z;
}
/* EOF */
```

Several finite difference methods were implemented on a rectangular lattice. These all proved to be less effective than the Euler method used along the characteristics (via the customized lattice, that is, using the characteristics to build the grid). The most effective of these schemes was the Lax-Wendroff scheme, as implemented in the following program. Mitchell and Griffiths [14, Chapter 4] give a good discussion of the Lax-Wendroff scheme and hyperbolic equations in general; see also Strikwerda [18].

A solution plotted from this program is presented in Figure 3. It is fairly similar to the plots in Figures 1 and 2 for the same parameter values using the other methods investigated, but it suffers from typical Gibbs phenomena, and does not maintain the sharp discontinuities actually present in the true solution along each characteristic. However, for the parameter values given, these discontinuities decay exponentially to zero, and for longer times the solutions of all three methods agree very well. In cases where a discontinuity does not decay to zero (but instead to a finite size between 0 and 1, as in Figures 4 and 5), the Lax-Wendroff method is visibly worse because the discontinuity is smeared out over several grid points. Away from the discontinuity agreement is good.

```
/* LAX-WENDROFF SCHEME
This is only part of a program.

Use of Lax-Wendroff scheme for a simple hyperbolic system
   du/dt =  du/dx + f(u,v),   f(u,v) = q1 (v-u) + r1 u(u-1);
   dv/dt = -dv/dx + g(u,v),   g(u,v) = q2 (u-v) + r2 v(v-1);
with Heaviside initial data
   u(0,x) = v(0,x) = 1 for x >  0,
                   = 0 for x <= 0.
After the nth step, u[k]=uu[k+500] represents u(n*lambda*h, k*h), where lambda
is the size of the time step divided by the size of the space step (which should
```

```
be below 1 for most of these schemes).

nu denotes the next u-array, that is, u one time step later.

We consider the values -480 <= k <= 480; 0 <= n <= 480 (= bign), with printouts
every 80 (=gap) steps.                                                        */

#define gap  80   #define bign 480

double q1, q2, r1, r2, t;
int a, b, n; /* a and b control the region of the array in which calculation is
                performed, n is the current number of the time steps          */

double uu[1001], vv[1001], nuu[1001], nvv[1001];
double  *u =  &uu[500]; double *v  =  &vv[500];
double *nu = &nuu[500]; double *nv = &nvv[500];

int k;           /*    to be used as a counter variable      */
double h; double lam;  /* lambda = time-step divided by space-step */

void Solve(void);      /* calls up Display when appropriate  */
double Trim(double z); /* ensures values stay in [0,1]       */
void LaxWen(void);     /* Lax-Wendroff method                */

void Solve(void)
{
      /* Initialize */
  for(k=-485; k<= 0; k++) {u[k] = 0.0; v[k] = 0.0;}
  for(k=1; k<= 485; k++)  {u[k] = 1.0; v[k] = 1.0;}
  for(n=1; n<=bign; n++) {
      a = 1 - n; b = n;
      for(k=a; k<= b; k++) {LaxWen();}
      }
       /* Stabilize */
   for(k=a; k<=b; k++)
      {u[k] = Trim(nu[k]); v[k] = Trim(nv[k]);}

      /* Display */
      if (n % gap == 0 ) {t = n * h * lam;Display();}
  }
}

void LaxWen(void)
{ nu[k] = u[k] + 0.5 * lam * (u[k+1] - u[k-1])
  + 0.5*lam*lam*(u[k+1] + u[k-1] - 2*u[k]) + h*lam*f(u[k],v[k]);
  nv[k] = v[k] - 0.5 * lam * (v[k+1] - v[k-1])
```

```
  + 0.5*lam*lam*(v[k+1] + v[k-1] - 2*v[k]) + h*lam*g(u[k],v[k]);
}

double Trim(double z)
{ if (z>1.0) return 1.0
  else if (z<0.0) return 0.0;
  else return z;
}
/* EOF */
```

3.1 Probabilistic simulation

The key to the probabilistic simulation is a simple recursive function called `Life`. This function tracks the path of an individual particle, updating the record of the left-most position yet reached by any particle each time the record is broken, and storing in arrays the time, place and type of each birth. Then, upon completion of a particle's run (since we only run each particle up until a pre-specified maximum time), we check to see if we have any more particles to do, and run the `Life` function on them.

Each particle is dealt with in a series of segments (the function `DoSegment`). An exponential random variable is generated (by `Rexp`)— the length of time until *either* the particle splits into two *or* the particle changes type. The position of the particle is updated by simply adding the length of time multiplied by its speed in its type for the segment to the current position. If the length of the segment takes us past our maximum time, we have completed the life story for that particle, and move on to the next one (incrementing `c`, the number of our current particle). Then we determine which event it was that did actually occur — birth or mutation. For birth we call the function `Create`, which stores the time, position and type in the arrays `tt`, `xx` and yy respectively. We also increment the counter `n` to inform us there is one more particle to be dealt with later. We then do another segment. If we change type, then we flip the type variable `y` and do the next segment.

Once we reach a point where we have completed the life story of a particle and there are no more sets of birth information unused (i.e. `c` greater than `n`), we have finished the simulation run.

If one is interested in calculating solely, for example, the left-most particle position, much calculation can be saved. Since a particle can only travel at speeds 1 and -1 we have immediate bounds on its future position (and identical bounds on all its future descendants). Thus, if the maximum time we are running until is T and the current record for left-most position is X, we can discard any particle in the simulation (denoting its position and time by (t, x)) for which $x - (T - t) > X$; particles satisfying this inequality have no chance of changing the record. The fact that their descendants also cannot break the record means we can discard the parent and save even generating the descendants.

```
/*  SIMULATION OF BRANCHING PROCESS MODEL

This is only part of a program. It shows how to extract information on the
left-most particle from the simulation (into the arrays leftu and leftv), and
```

```
save some computation if this is all we are interested in (by pruning away
particles far away from the left-most).

While in type 1 particle moves at speed b[1] (wlog set to 1), mutates at rate
q[1] and breeds at rate r[1] - in type 2 particle moves at speed b[2] (wlog set
to -1), mutates at rate q[2] and breeds at rate r[2] - q and r user inputs.

The program notes the position of the left-most particle in each of NUMRUN (1000)
simulations starting from 1 particle at the origin (doing 1000 runs for that
particle being type 0, and 1000 for type 1), observing each simulation at T (6)
points, each GAP (user input) units apart.                                    */

#define NUMRUN 1000
#define MAXPART 1000000 /* limit on particle numbers, warning if exceeded      */
#define T 6
double b[3],Lpos[T+1],tt[MAXPART+1],xx[MAXPART+1],x,GAP,leftu[T+1][NUMRUN],
       leftv[T+1][NUMRUN],q[3],r[3];
int y,c,yy[MAXPART+1],TYPE,k;

void OneRun(void);
void DoSegment(void);
void Life(void);
void Create(void);

int main()
{
  int i,j; b[1]=1.0; b[2]=-1.0;
/* Simulations starting from 1 particle of type 1 */
  TYPE=1;
  for(i=0; i<NUMRUN; ++i) {
    OneRun();
    for(j=0;j<=T;++j) {
      leftu[j][i]=Lpos[j];
    }
  }
/* Simulations starting from 1 particle of type 2 */
  TYPE=2;
  for(i=0; i<NUMRUN; ++i) {
    OneRun();
    for(j=0;j<=T;++j) {
      leftv[j][i]=Lpos[j];
    }
  }
}
void OneRun(void)
{
```

```
  int i;

  n=0; c=0; t=0.0; x=0.0; Lpos[0]=x; tt[0]=t; xx[0]=x;
  yy[0]=TYPE;
  for(i=1; i <= T; ++i) {
    Lpos[i]=10000;
  }
  Life();
}
double RUnif() /* a different random number generator could go here */
{
  double drand48();
  return(drand48());
}
double Rexp(lam)
double lam;
{
  double log();
  return(-log(RUnif())/lam);
}
void DoSegment(void) /* update particles age and position */
{
  double e,Rexp();
  int i,j;

  e=Rexp(q[y]+r[y]); j=t/GAP; k=(t+e)/GAP;
  if (k > T)
    k=T;
  for(i = j + 1; i <= k; ++i) {
    if (Lpos[i] > x + b[y] * (i * GAP - t)) {
      Lpos[i]=x+b[y]*(i*GAP-t); /* this notes any new records
                                   set by this particle */
    }
  }
  x += e*b[y]; t += e;
}
void Life(void)
{
  double RUnif();

    /* initialise next particle */
  t = tt[c]; x = xx[c]; y = yy[c];

  while (t < T*GAP && (x+t-(T*GAP)) < Lpos[T]) {
    DoSegment();
    if (RUnif() < (r[y]/(q[y]+r[y])))
```

```
      Create();
    else
      y = 3 - y; /* This sends 1 to 2 and 2 to 1 */
  }
  c++;
  if (c >= MAXPART)
    printf("\nFilled up particle arrays\n");
  else if (c <= n)
    Life();
}
void Create(void) /* a new particle has been born,
                       store its details */
{
  n++;
  if (n < MAXPART)
     {tt[n]=t; xx[n]=x; yy[n]=y;}
}
/* EOF */
```

Plots of the solutions obtained by simulation are presented in Figures 2 and 5. These agree very well with those produced by the Euler method for corresponding parameters in Figures 1 and 4. When more simulation is done the probabilistic plots are smoother and agreement is even better. This simulation method could of course easily be extended to the n-type case.

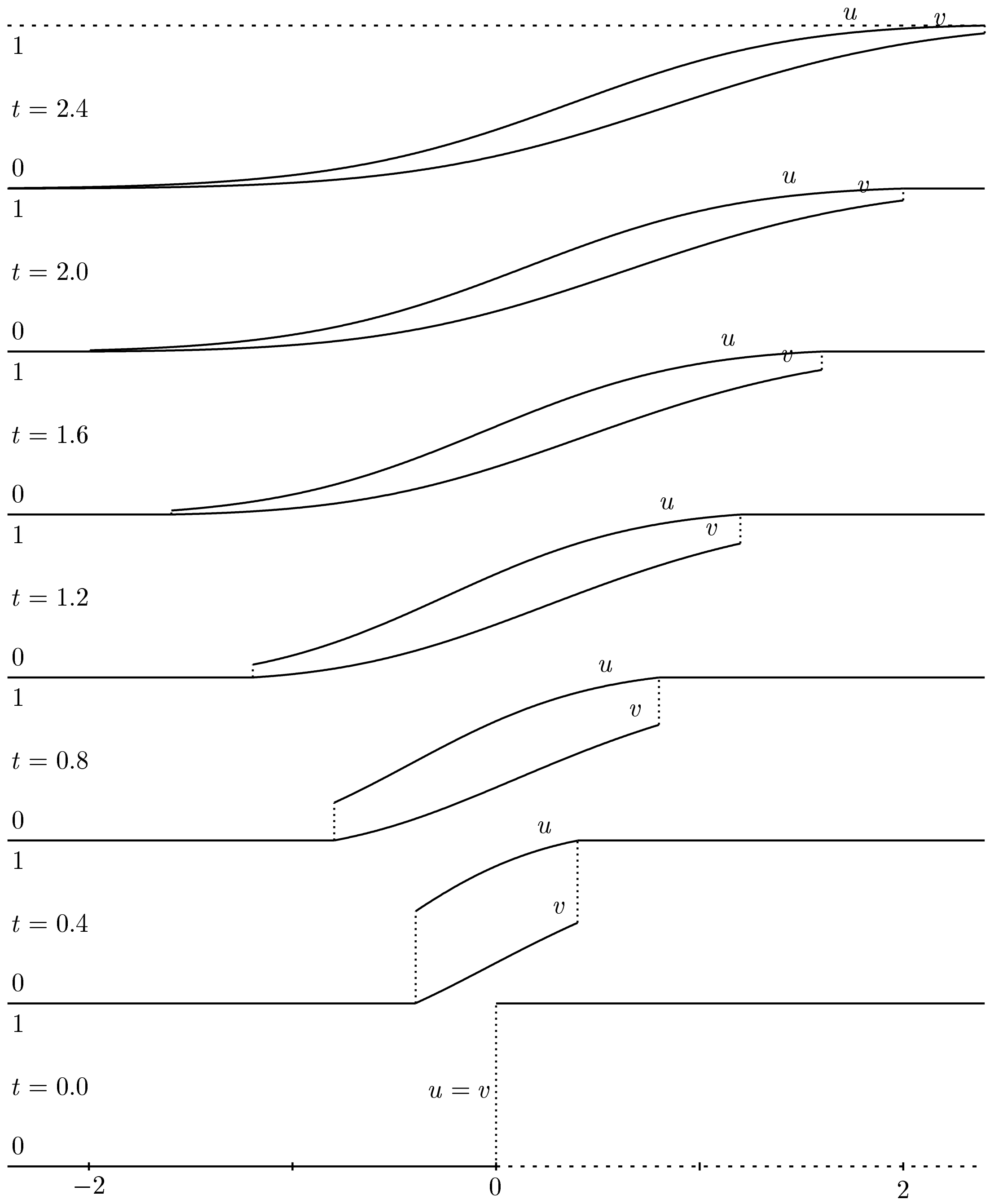
3.2 Discussion of figures

Consider the situation in which there is just one particle at time 0, of type 2 and with position x. We know that $v(t,x)$ is the probability that all particles are to the right of 0 at time t. Since no particle can travel left at speed greater than 1, $v(t,x)=1$ for $x>t$. However, if $x=t$, then $v(t,t)$ is the probability that up to time t our initial particle has no line of descent consisting only of particles of type 2: in other words, that every descendant of our initial particle spends some time before t moving right (in which case it can never get to 0 at time t). It is therefore clear that there is a positive jump $1-v(t,t)$ in $v(t,\cdot)$ at time t, and that this jump may be calculated by regarding any particle of type 1 as 'dead' and ignoring it and its descendants. Precisely, the jump $1-v(t,t)$ is the probability that a continuous-time branching process starting from 1 particle, and with birth-rate r_2 and death-rate q_2 (per individual) survives until time t. If $q_2>r_2$, the situation in Figures 1, 2 and 3, then this survival probability tends exponentially fast to 0. If $r_2>q_2$, the situation in Figures 4 and 5, then $v(t,t)\to 1-\pi$, where the extinction probability π satisfies

$$\pi=\frac{q_2}{q_2+r_2}+\frac{r_2}{q_2+r_2}\pi^2,$$

so that $\pi=q_2/r_2$.

Two other features of the pictures are worthy of comment. Firstly, the fast convergence to the travelling wave in Figures 4 and 5 illustrates the case discussed in section 2.4. Secondly, the almost linear nature of $v(t,\cdot)$ for small t, clearly apparent in Figures 1–3 may be explained as

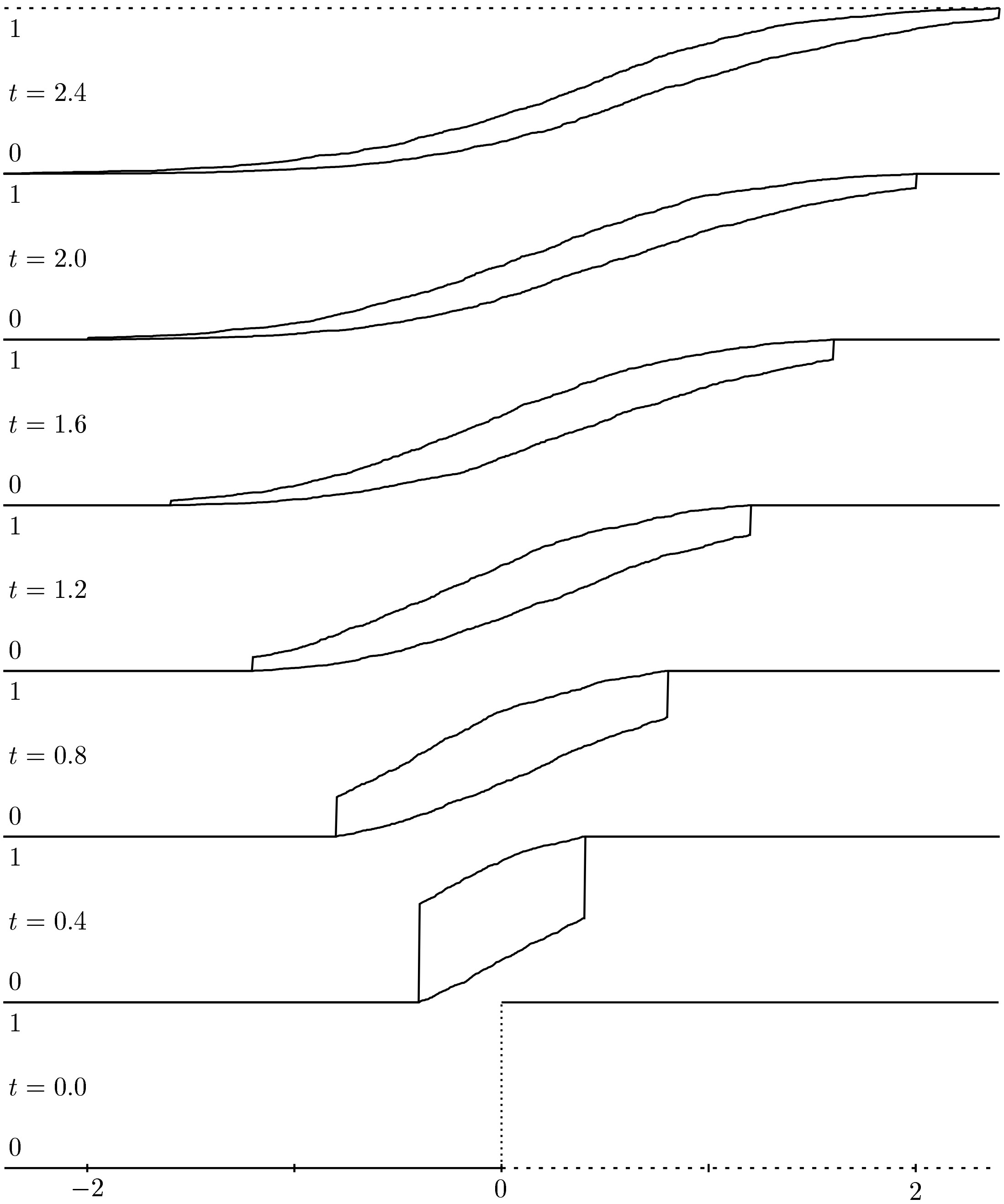


$q_1 = 1.00, \quad q_2 = 2.00, \quad r_1 = 2.00, \quad r_2 = 1.00.$

Graphs of u and v for $t = 0.0(0.400)2.400$

Method: Euler2 with $h = 0.0025$

Figure 1: Numerical solution calculated using modified Euler method along the characteristics

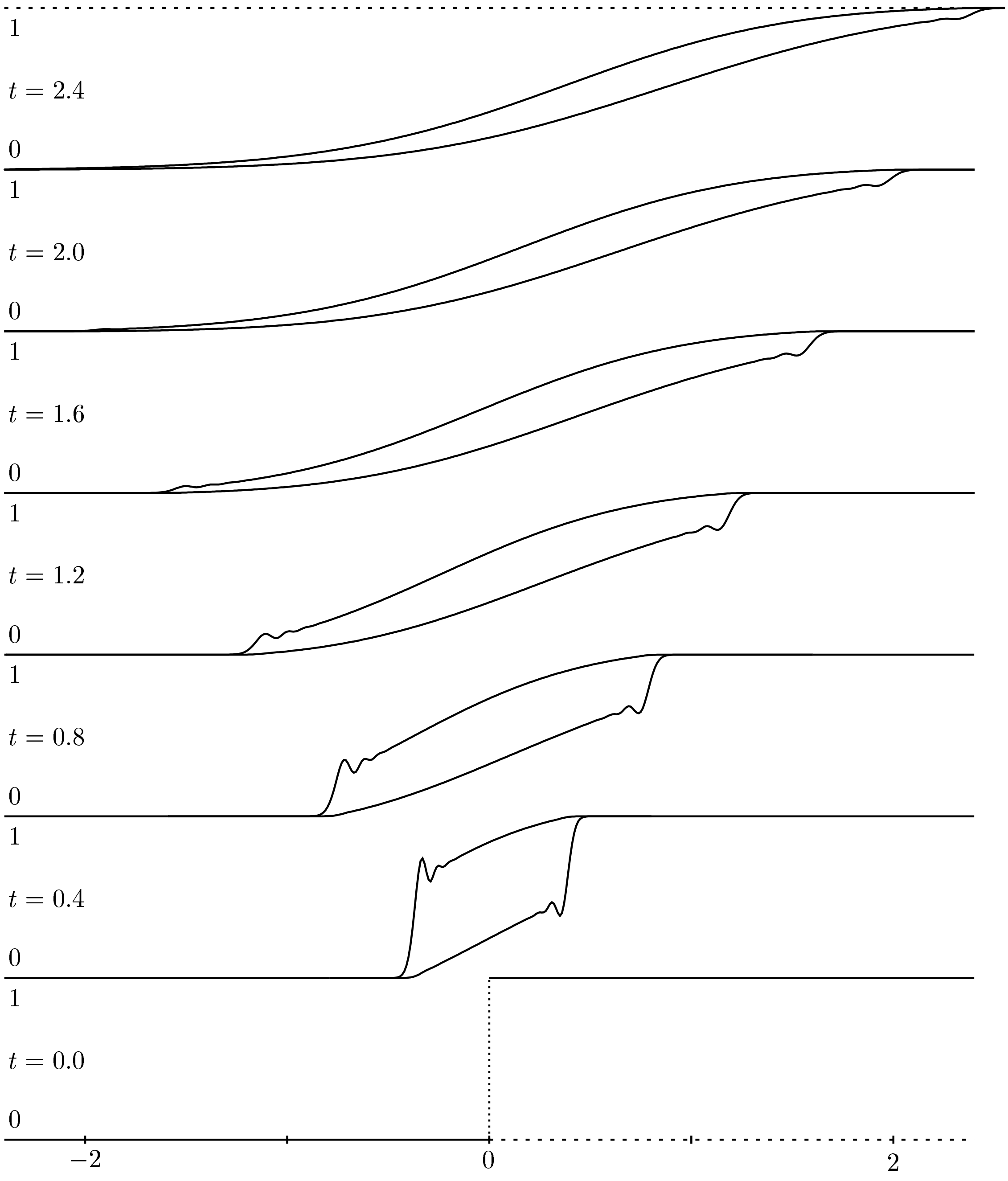


$q_1 = 1.00, \quad q_2 = 2.00, \quad r_1 = 2.00, \quad r_2 = 1.00.$

Distribution of left-most particle for $t = 0.0(0.400)2.400$

1000 Runs each from type 1, and from type 2, initial particle

Figure 2: Numerical solution calculated from probability simulation

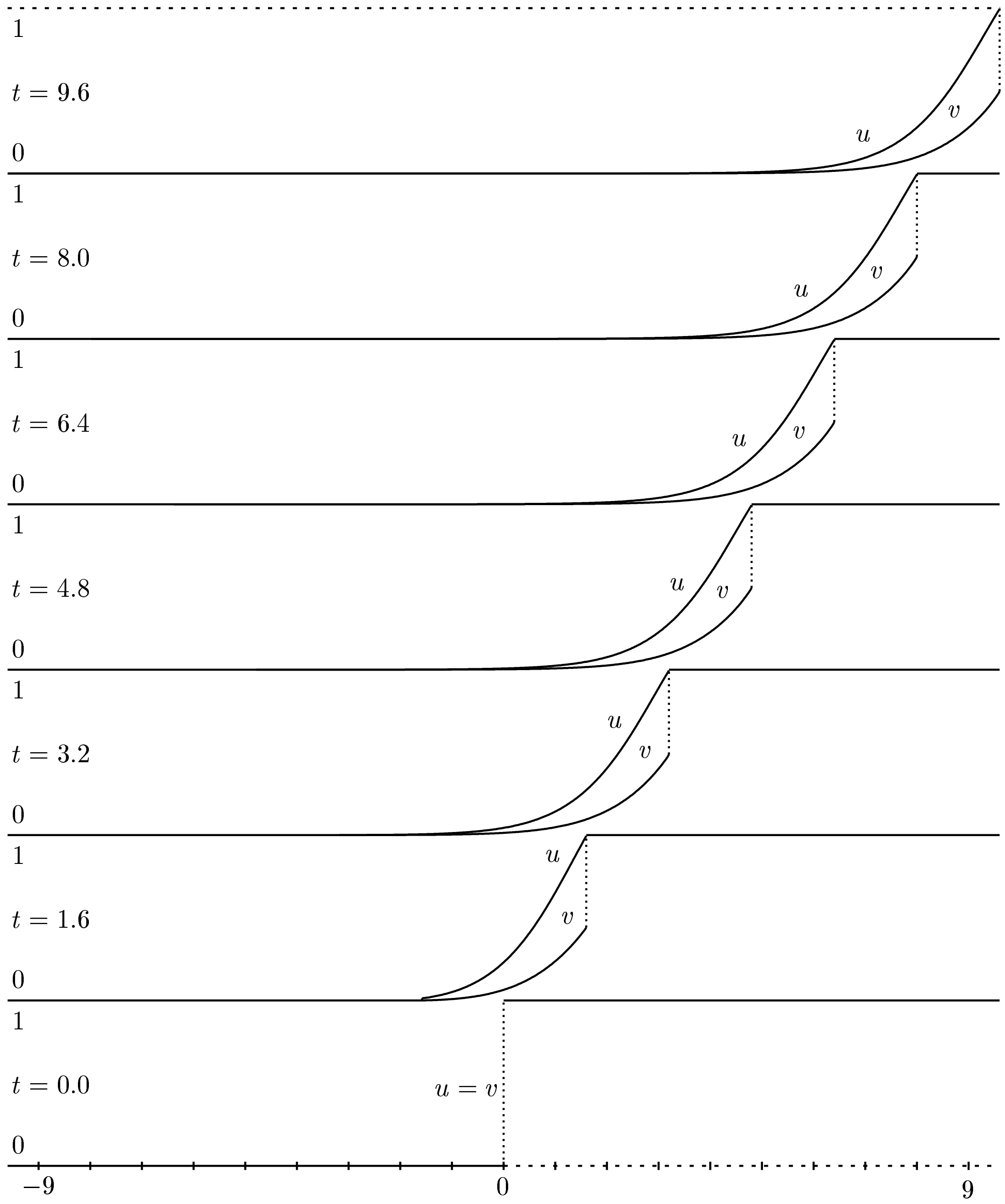


$q_1 = 1.00, \quad q_2 = 2.00, \quad r_1 = 2.00, \quad r_2 = 1.00.$

Graphs of u and v for $t = 0.0(0.400)2.400$

Method: Lax-Wendroff with $h = 0.010$, $\lambda = 0.5$

Figure 3: Numerical solution calculated using Lax-Wendroff method

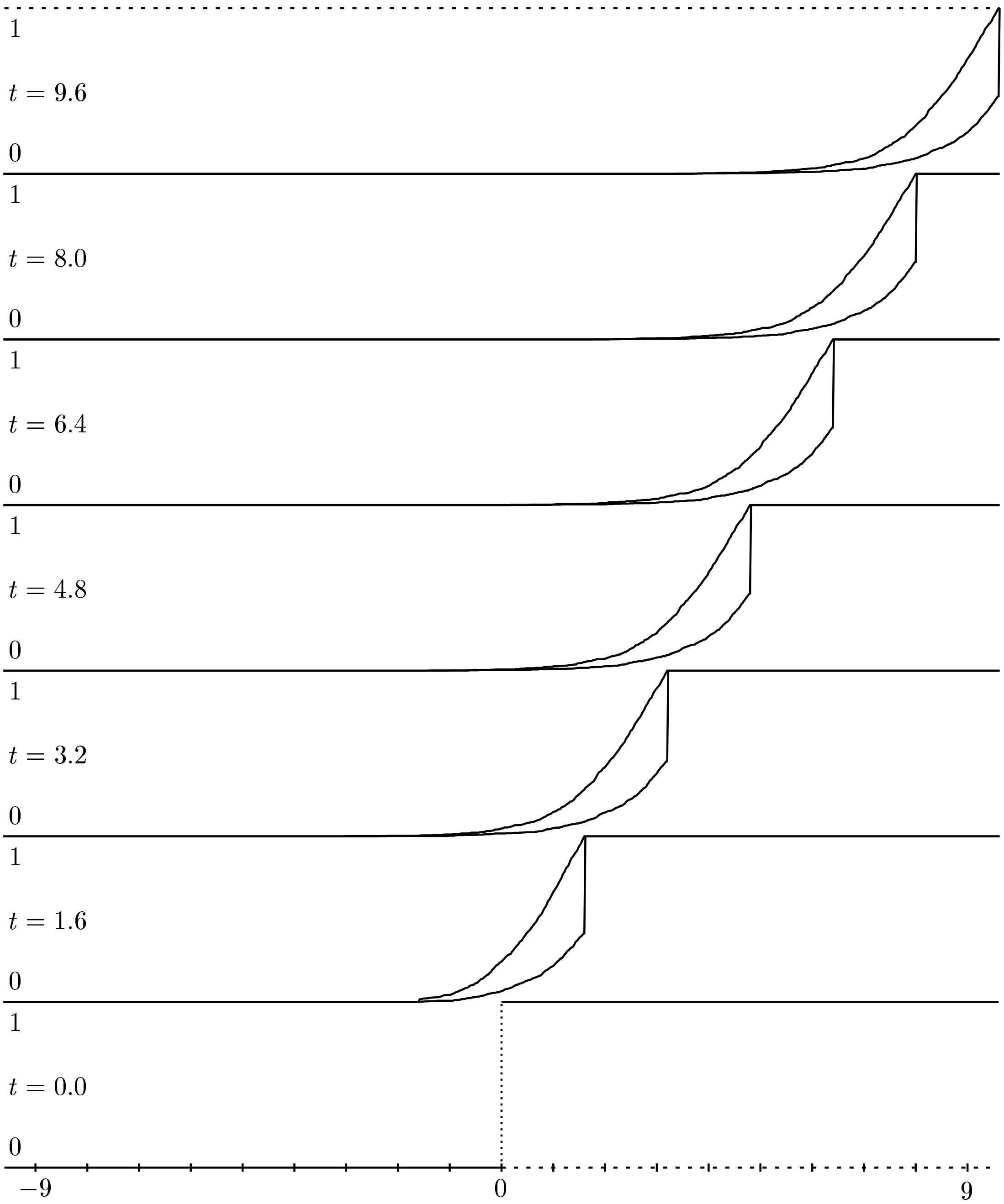


$q_1 = 2.00, \quad q_2 = 1.00, \quad r_1 = 1.00, \quad r_2 = 2.00.$

Graphs of u and v for $t = 0.0(1.600)9.600$

Method: Euler2 with $h = 0.0100$

Figure 4: Numerical solution calculated using modified Euler method along the characteristics



$q_1 = 2.00, \quad q_2 = 1.00, \quad r_1 = 1.00, \quad r_2 = 2.00.$

Distribution of left-most particle for $t = 0.0(1.600)9.600$

1000 Runs each from type 1, and from type 2, initial particle

Figure 5: Numerical solution calculated using probability simulation

follows. Again, consider the situation in which there is just one particle at time 0, of type 2 and with position x. For small t, the dominant contribution to $v(t,x)$ will arise from cases where there is a random time S before t at which the particle changes type. Thus the particle moves left for a time S and right for a time $t-S$, ending up at $x-S+t-S$. Thus, for small t,

$$v(t,x) \approx \mathbb{P}(x-S+t-S>0) = \mathbb{P}\Big\{S < \frac{1}{2}(x+t)\Big\} = 1 - e^{-\frac{1}{2}q_2(x+t)} \approx \frac{1}{2}q_2(x+t).$$

Acknowledgements. Owen Lyne would like to thank the EPSRC for financial support in the form of a Research Studentship.

References

[1] Bramson, M. D. (1983) Convergence of solutions of the Kolmogorov equation to travelling waves. *Mem. Amer. Math. Soc.*, **285**, 1–190.

[2] Champneys, A., Harris, S., Toland, J., Warren, J. & Williams, D. (1995) Algebra, analysis and probability for a coupled system of reaction-diffusion equations. *Phil. Trans. R. Soc. Lond.*, A **350**, 69–112.

[3] Dawson, D. A. (1993) Measure-valued Markov processes, *École d'Été de Probabilités de Saint Flour, 1991*, Lecture Notes in Mathematics, **1541**. New York: Springer-Verlag.

[4] Dunbar, S. R. (1988) A branching random evolution and a nonlinear hyperbolic equation. *SIAM J. Appl. Math*, **48**, 1510–1526.

[5] Dynkin, E. B. (1994) *An Introduction to Branching Measure-Valued Processes.* Providence, Rhode Island: American Mathematical Society.

[6] Fisher, R. A. (1937) The wave of advance of an advantageous gene. *Ann. Eugenics*, **7**, 353–369.

[7] Hadeler, K. P. (1995) Travelling fronts in random walk systems. *Forma*, **10**, 223–233.

[8] Holmes, E. E. (1993) Are diffusion models too simple? A comparison with telegraph models of invasion. *Amer. Nat.*, **142**, 779–795.

[9] Kolmogorov, A. N., Petrowski, I. & Piscounov, N. (1937) Étude de l'équation de la diffusion avec croissance de la quantité de matière et son application à un problème biologique. *Mosc. Univ. Bull. Math.*, **1**, 1–25.

[10] Lyne, O.D. (2000) Travelling waves for a certain first-order coupled PDE system. *Electron. J. Probab.*, **5**:14, 1–40.

[11] Lyne, O.D. (1996) Probability and analysis for a hyperbolic coupled PDE system. Unpublished Ph.D. thesis, University of Bath.

[12] McKean, H. P. (1975) Application of Brownian motion to the Equation of Kolmogorov-Petrovskii-Piskunov. *Comm. Pure Appl. Math.*, **28**, 323–331.

[13] McKean, H. P. (1976) Correction to the above. *Comm. Pure Appl. Math.*, **29**, 553–554.

[14] Mitchell, A. R. & Griffiths, D. F. (1980) *The Finite Difference Method in Partial Differential Equations.* Chichester: Wiley

[15] Neveu, J. (1987) Multiplicative martingales for spatial branching processes. *Seminar on Stochastic Processes* (ed. E. Çinlar, K. L Chung and R. K. Getoor), Progress in Probability and Statistics **15**. pp. 223–242. Boston: Birkhäuser.

[16] Othmer, H. G., Dunbar, S. R. & Alt, W. (1988) Models of dispersion in biological systems. *J. Math. Biol.*, **26**, 263–298.

[17] Rogers, L. C. G. & Williams, D. (1987) *Diffusions, Markov Processes and Martingales, Volume 2: Itô Calculus.* Chichester: Wiley.

[18] Strikwerda, J. C. (1989) *Finite Difference Schemes and Partial Differential Equations.* Pacific Grove: Wadsworth & Brooks/Cole