Il difetto sta - per usare la felice immagine di Jerome S. Bruner - nel trascurare gli aspetti di pertinenza della "mano sinistra" (fantasia, intuizione, arte, gioco) disseccando tutto nella mera sistemazione razionale (di pertinenza della "mano destra"). Ed è notevole merito del Bruner (psicologo, non matematico) di aver ravvisato nella matematica, e nell'insegnamento e apprendimento della matematica, il campo ove l'intervento della "mano sinistra" sarebbe particolarmente vivificatore ed è invece purtroppo pressoché totalmente assente.

[Bruno De Finetti, 1971]

0. I lavori presentati sono esperienze e proposte didattiche condotte e progettate con la "mano sinistra", o almeno questa è l'intenzione che li ha ispirati. Ma il posticipare il ricorso a strumenti di indagine concettuale razionali e formali rispetto ad un primo approccio esplorativo tipico del gioco non è l'unico tratto che li accomuna.

1. È ormai opinione diffusa che il XX secolo, rispetto al secolo che lo ha preceduto, ha evidenziato - nonostante l'indubbio successo tecnologico delle scienze applicate - i limiti storici ed epistemologici delle scienze esatte e naturali, non escluso il linguaggio matematico sul quale esse sono logicamente fondate.

    1.1. Il riconoscimento della irriducibile complessità e della intrinseca imprevedibilità di molti fenomeni naturali e artificiali, che risultano modellizzabili esclusivamente esplicitando le relazioni non lineari e ricorsive tra i processi che li compongono, ha percorso la strada - aperta all'inizio del secolo da Poincaré - delle ricerche nel campo del cosiddetto "caos", la cui descrizione geometrica è quella degli oggetti frattali introdotti da Mandelbrot e considerati in precedenza forme patologiche. Se ancora nell'anno 1900 il matematico Hilbert sosteneva la decidibilità in linea di principio di tutti i problemi matematici con "un numero finito di processi puramente logici", i risultati ottenuti da Gödel, Turing, Church e Tarski minano i fondamenti stessi di questa concezione della Matematica e ne permettono, al tempo stesso, una reinterpretazione meno dogmatica.

    1.2. Questa "rivoluzione scientifica" (Kuhn, 1962) dovrebbe forse restare al di fuori dai contenuti e dallo stile di insegnamento della scuola secondaria? Fortunatamente non è questo lo spirito dei programmi elaborati dalla commissione Brocca che - oltre a ridefinire metodi e contenuti - introducono anche negli indirizzi scientifico-tecnologici l'insegnamento

della filosofia come occasione di riflessione critica sulla conoscenza scientifica e matematica.

1.3. Gli argomenti dei lavori presentati sono esempi di questo nuovo "paradigma scientifico" (Kuhn, 1962): il caos sorprendente della dinamica di semplici sistemi deterministici e non lineari, la geometria generativa delle complesse forme frattali, l'evoluzione computazionalmente imprevedibile di alcuni automi cellulari. Per quanto possano apparire esotici o addirittura esoterici rispetto agli argomenti che tradizionalmente sono oggetto di insegnamento delle discipline scientifiche e della matematica in particolare, essi sono intimamanete legati ad alcuni concetti più ortodossi e comunque esplicitamente previsti dai nuovi programmi, come l'implementazione e l'analisi della convergenza di algoritmi numerici iterativi e ricorsivi, le trasformazioni geometriche affini del piano, le grammatiche generative, l'approssimazione numerica della soluzione di equazioni differenziali e la decidibilità computazionale di un problema.

2. Premesso che le proposte sono da interpretarsi come singole esperienze di "rottura" di una pratica didattica più tradizionale e continuativa, è comunque importante che a questa decisa innovazione sul piano dei contenuti nel processo di insegnamento e apprendimento corrisponda anche una rilevante innovazione metodologica.

2.1. La visione auspicata è quella di un rapporto dialogico tra analisi e sintesi nel contesto del quale il computer ed alcuni strumenti software aventi valenza generale (un linguaggio di programmazione general-purpose per la realizzazione di semplici simulazioni, un ambiente interattivo e programmabile per il calcolo numerico e simbolico, un foglio di dati e di calcolo) costituiscano un laboratorio per lo svolgimento di veri e propri esperimenti virtuali. In questa visione la risorsa informatica è uno strumento di esplorazione di modelli della realtà fenomenica naturale e artificiale che ha le proprie caratteristiche peculiari. Il ricorso esclusivo a software con funzionalità molto generali permette di acquisire conoscenze relative allo strumento contemporaneamente al suo uso come mezzo di indagine, realizzando un'integrazione - troppo spesso trascurata - tra matematica, fisica ed informatica, ma senza trasformare il computer da "strumento" ad "oggetto" di conoscenza.

2.2. Nell'interpretazione della realtà l'insegnamento tradizionale privilegia un approccio sintetico e la deduzione formale di conseguenze da un modello definito a priori. L’approccio proposto inverte questa prospettiva evidenziando l'importanza di una fase di analisi che si concretizza nella realizzazione di un programma di simulazione o nell'esplorazione

sperimentale di un modello "informatico", prima ancora che "matematico". Non si intende sottovalutare l'aspetto sintetico dei modelli matematici tradizionali, ma si ritiene interessante per lo studente confrontare due diversi metodi di lettura del "libro della natura" (Galilei, citato nella premessa ai nuovi programmi di insegnamento della matematica elaborati dalla commissione Brocca).

2.3. Il ricorso alle funzionalità di ambienti software come Mathematica o Maple - oltre a fornire allo studente alcune conoscenze relative a strumenti sempre più diffusi ed utilizzati - permette di prendere in considerazione alcuni modelli interessanti sotto l'aspetto del loro comportamento e strutturalmente analoghi a quelli comunemente argomento di lezione, ma che richiedono tecniche di trattamento, numeriche o simboliche, non disponibili allo studente della scuola superiore. Molta ricerca, perfino nell'ambito della matematica astratta, viene oggi condotta con questi strumenti e con queste metodologie che ritengo validi e trasferibili anche nel campo della didattica.

2.4. Tutte le forme di conoscenza umana - non esclusa la conoscenza scientifica e la formalizzazione stessa della matematica - presentano aspetti epistemologici che non dovrebbero essere trascurati: ritengo importante inserire nella proposta didattica, eventualmente in collaborazione con l'insegnamento di filosofia, una fase di valutazione critica della (complessa) relazione tra realtà fenomenica e modelli analitici e sintetici di interpretazione che includa considerazioni di carattere storico e sociale.

3. Nei lavori proposti particolare attenzione è stata posta nel selezionare argomenti, modalità e contesti di presentazione interdisciplinari capaci di stimolare la curiosità esplorativa degli studenti come se si trattasse di partecipare ad un "gioco di idee". Questa cura è, a mio avviso, particolarmente evidente nella prima delle presenti proposte che è già stata oggetto di sperimentazione didattica.

Studio del comportamento caotico e frattale di una semplice funzione

Giorgio Meini

In memoria di Ian Malcom
(il vero protagonista di JURASSIC PARK)

Come Ian Malcom ci ricordava la teoria del caos nasce dal fallimento degli studi nel campo delle previsioni meteorologiche a lungo termine: nel 1963 il meteorologo Edward Lorenz formulò il primo modello di "dinamica caotica"

> *Ian Malcom era uno dei più famosi rappresentanti di quella nuova generazione di matematici che mostravano un vivo interesse per "i meccanismi del mondo reale". Questi studiosi, sotto molti e fondamentali aspetti, avevano rotto la tradizione di isolamento dei matematici. Per prima cosa si servivano continuamente del computer, cosa che i matematici tradizionali non vedevano di buon occhio. Poi lavoravano quasi esclusivamente con equazioni non lineari, nel campo emergente del cosiddetto caos. Terza cosa, sembravano voler fare tutto il possibile affinché i loro sistemi matematici descrivessero qualcosa che di fatto esisteva nel mondo reale.*
>
> [Michael Crichton, *Jurassic Park*, III ed. it., Garzanti, 1993, pag. 98]

Nello spettacolare film di Steven Spielberg lo stravagante e profetico matematico Malcom sopravvive al morso del *Tyrannosaurus rex*, ma nel romanzo originale di Crichton [1], di cui Malcom è indubbiamente il protagonista, la sua agonia prosegue fino alla morte attraverso un lungo e lucido delirio in cui denuncia l'inconsistenza della pretesa fondamentale della scienza moderna: prevedere e controllare i fenomeni del mondo reale.

"- La teoria del caos manda a gambe all'aria tutto questo. Sostiene che, in certe situazioni, nulla è prevedibile. Non si può prevedere il tempo, se non nell'arco di pochi giorni. Bisogna gettare la spugna. Non è fattibile. [...]. E` un'impresa da pazzi. E` vana quanto cercare di trasformare il piombo in oro. Oggi gli sforzi degli alchimisti ci fanno ridere perché sappiamo che il loro intento era impossibile. Ma generazioni future guarderanno a noi e rideranno nello stesso modo. [...]. Perché di fatto vi sono enormi categorie di fenomeni che sono intrinsecamente imprevedibili -
- Questo è quanto sostiene la teoria del caos? -
- Sì, ed è straordinario quante poche siano le persone che vi prestino ascolto -, disse Malcom"
[Michael Crichton, *Jurassic Park*, III ed. it., Garzanti, 1993, pag. 203]

I computer di "Jurassic Park" non sono solo i Silicon Graphics con cui la Magic Light ha ricreato per Spielberg dinosauri virtuali, ma estremamente realistici. Il film, come il libro, è una intelligente denuncia della mancanza di consapevolezza rispetto ai limiti intrinseci delle tecniche di controllo e dei modelli di previsione. Si parte dall'ingegneria genetica: l'improbabile catena di eventi che nella storia permette di giungere alla clonazione di animali estinti da milioni di anni - il rinvenimento di DNA di dinosauro nel sangue contenuto in zanzare fossilizzatesi nell'ambra - è una tecnica narrativa per illustrare i rischi connessi a molte pratiche di questa disciplina. Si arriva al controllo computerizzato dei sistemi complessi: George Orwell, con la metafora del "Grande Fratello" in *1984*, si riferiva ai rischi sociali connessi alla possibilità tecnologica (informatica e telematica) di acquisire, memorizzare, organizzare e trasmettere enormi quantità di informazioni; Crichton invece, pur sottolineando questo aspetto (la InGen utilizza tre supercomputer Cray XMP per ricostruire i filamenti di DNA deteriorati dal tempo), costruisce la catastrofica successione di eventi di "Jurassic Park" a partire dai ripetuti fallimenti del sistema di controllo. Il montaggio del film può lasciare allo spettatore il dubbio che la catastrofe avvenga a causa del tradimento da parte dell'odioso programmatore Nedry, ma il libro è esplicito nel ricordare che un sistema dal comportamento imprevedibile non può comunque essere controllato: alcuni dinosauri sono infatti fuggiti dall'isola alla volta della terraferma molto tempo prima del week-end nel quale si svolge la storia e sopravviveranno agli eventi.

Computer Programming n. 32, Gennaio 1995, pp. 49-54

Come Malcom ci ricordava, la teoria del caos nasce dal fallimento degli studi nel campo delle previsioni meteorologiche a lungo termine: nel 1963 il meteorologo Edward Lorenz formulò il primo modello di "dinamica caotica" - un semplice sistema di equazioni differenziali non lineari descrivente il comportamento di un volume circoscritto di atmosfera e per il cui studio fu indispensabile (ma sufficiente!) un piccolo computer - concludendo "che lo sbattere di ali di una frafalla in Amazzonia può causare un tornado in Florida l'anno successivo": la dinamica (il comportamento) del modello era altamente dipendente dalle condizioni iniziali. Lo studio di un sistema, sia naturale che artificiale, porta alla formulazione di un modello matematico (per esempio un sistema di equazioni differenziali) che ne descrive il variare di alcune grandezze caratteristiche con il trascorrere del tempo, a partire da uno stato iniziale noto. Per utilizzare un modello matematico per la previsione o il controllo dell'evoluzione nel tempo di un sistema reale è necessario "misurare" i valori delle grandezze che caratterizzano lo stato iniziale, ma l'operazione di misura del valore di una grandezza fisica può essere effettuata solo con una precisione limitata. Un modello non lineare è tipicamente dipendente dalle condizioni iniziali nel senso che l'ineliminabile errore di misura che i valori di queste presentano impedisce di valutare l'evoluzione futura del sistema: a piccolissime variazioni di questi valori, conseguenze essenzialmente casuali del procedimento di misura, corrispondono infatti comportamenti del sistema del tutto diversi. Questa caratteristica è propria del modello e, spesso, del sistema oggetto di studio e non è dovuta ad un limite delle tecniche o delle risorse di calcolo impiegate per "trattare" il modello stesso (non è dovuta per esempio al fatto che, non disponendo di tecniche analitiche per risolvere il sistema di equazioni differenziali, si debba ricorrere a tecniche di approssimazione numerica sotto la forma di un programma per calcolatore).
Robert May è un biologo teorico che nei primi anni '70 ha incontrato problematiche analoghe studiando modelli matematici della crescita delle popolazioni animali. In un famoso articolo sull'argomento pubblicato su *Nature* nel 1976 ("Simple Mathematical Models with Very Complicated Dynamics") scriveva: "Vorrei che a chi studia matematica venisse sottoposta la seguente equazione:

$$y = x\left[(1+r) - \frac{r}{k}x\right]$$

Questa equazione può essere studiata da un punto di vista fenomenologico iterandola con un computer e il suo studio sarebbe di grande aiuto allo studente per capire i sistemi non lineari." Seguiamo dunque questo consiglio.
La crescita di una popolazione animale priva di predatori e che disponga di infinite risorse alimentari ed ambientali è descritta dall'equazione differenziale dx/dt=rx avente come soluzione la funzione esponenziale $x=x_0e^{rt}$ (r è il fattore di accrescimento e x_0 la popolazione iniziale), il cui grafico è riportato nella **figura 1** (realizzato con Derive della Soft Warehouse).

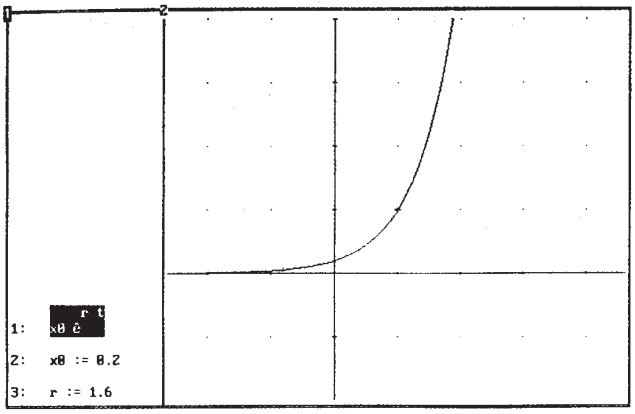


Figura 1 - La crescita di una popolazione animale priva di predatori e che disponga di infinite risorse alimentari ed ambientali.

Ma, nonostante l'economia umana ancora non lo riconosca, il concetto di "risorse limitate" è un principio fondamentale dell'ecologia moderna: l'equazione differenziale "logistica" **dx/dt=rx(1-x/k)**, introdotta da Lotka nel 1925, pur avendo un comportamento iniziale esponenziale rappresenta la limitatezza delle risorse disponibili per l'accrescimento con il parametro k (capacità portante). La soluzione dell'equazione logistica (calcolata con Derive)
è rappresentata nel grafico di **figura 2** (anche questo realizzato con Derive): è evidente come la popolazione x

$$x = \frac{k}{1-(1-\frac{k}{x_0})e^{-rt}}$$

evolva con il tempo verso uno stato di equilibrio corrispondente alla capacità portante k.

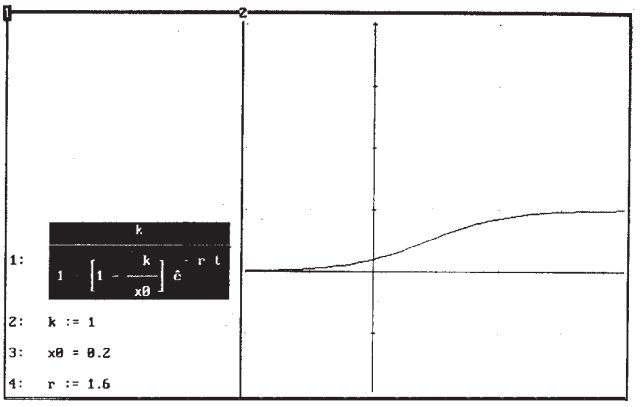


Figura 2 - La crescita di una popolazione animale priva di predatori e che disponga di limitate risorse alimentari ed ambientali.

Le soluzioni delle equazioni differenziali descrivono una evoluzione "continua" nel tempo - nel senso che rappresentano grandezze che variano istante per istante - del sistema che modellano. Ma l'accrescimento delle popolazioni avviene per generazioni, cioè in modo "discreto" rispetto al trascorrere del tempo: possiamo sostituire all'equazione differenziale logistica una equazione "alle differenze finite" che rappresenti ricorsivamente l'incremento

di popolazione rispetto alla generazione precedente:

$$\Delta x = r x_i\left(1 - \frac{x_i}{k}\right) \rightarrow x_{i+1} =$$
$$= x_i + r x_i\left(1 - \frac{x_i}{k}\right) = (1 + r)x_i - \frac{r}{k} x_i^2 =$$
$$= x_i\left[(1 + r) - \frac{r}{k} x_i\right]$$

Allo stesso risultato era giunto Verhulst nel 1845 stabilendo il tasso di crescita di una popolazione a partire dall'incremento inter-generazionale $r=(x_{i+1}-x_i)/x_i$. Se il tasso di crescita si mantenesse costante nel tempo (risorse infinite) si avrebbe $x_{i+1}=x_i+rx_i=(1+r)x_i$, cioè $x_i=(1+r)^i x_0$: è il meccanismo di crescita esponenziale. Ma, dato che le risorse sono finite, si avrà un massimo possibile di popolazione k: il tasso di crescita r deve diminuire quando la popolazione si avvicina al valore limite k e divenire 0 quando raggiunge tale limite, potrebbe quindi avere la forma $r-cx_i$ (c costante); dato che r-ck=0 si ha c=r/k e l'equazione ricorsiva diviene

$x_{i+1} = (1 + r - cx_i)\, x_i = (1 + r)\, x_i - (r/k)\, x^{i2} = x_i[(1 + r)-(r/k)\, x_i]$.

In entrambi i casi si ottiene l'equazione "consigliata" da May in forma ricorsiva. Una equazione ricorsiva può essere studiata iterandone il calcolo a partire dal valore della popolazione relativo alla generazione iniziale x_0 fino a quello relativo alla generazione N-esima utilizzando di volta in volta il risultato come valore di feed-back; il metodo è esemplificato dal seguente frammento di programma in linguaggio C:

```
x=x0;
for (i=1; i<=N; i++)
x=x+r*x*(1-x/k);
```

Il programma *iter* (**listato 1**), costruito sulla base di questo ciclo, costituisce un semplice esempio di simulazione algoritmica di un fenomeno reale e produce come risultato un file in un formato leggibile da parte di *Mathematica* (si veda il riquadro al termine dell'articolo per una breve descrizione).

Con *Mathematica* è possibile realizzare rappresentazioni grafiche delle liste di valori (corrispondenti alle densità di popolazione relative alle successive generazioni) prodotte dal programma *iter*.

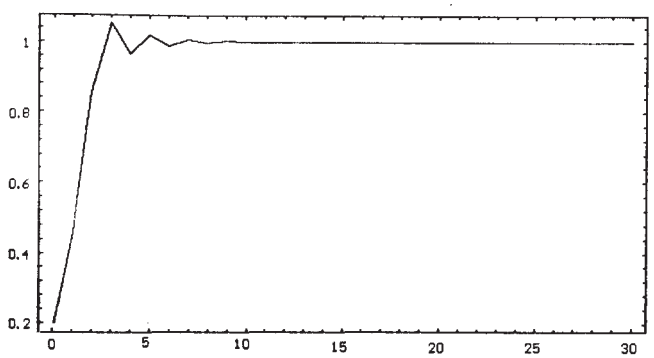


Figura 3 - L'equazione logistica per $x_0 = 0.2$, $r = 1.6$ e $k = 1.0$.

Ci si aspetta una semplice "discretizzazione" del grafico della **figura 2** e il grafico riportato nella **figura 3** ottenuto stabilendo i valori $x_0=0.2$, $r=1.6$ e $k=1.0$ - valori plausibili se x rappresenta la densità di popolazione rispetto ad una fissata estensione di territorio anziché la popolazione assoluta - sembrerebbe infatti confermare l'andamento "logistico": una rapida crescita fino alla densità di equilibrio stazionario con le risorse effettivamente disponibili.

LISTATO 1

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

void main (int argc, char *argv[])
{
 FILE *fd;
 double x,x0,r,k=1.;
 int i,N;

 if (argc!=4)
   {
    printf("\nUso: ITER x0 r N");
    return;
   }

 x0=atof(argv[1]);
 if (x0==0.0)
   {
    printf("\nValore di x0 non valido");
    return;
   }

 r=atof(argv[2]);
 if (r==0.0)
   {
    printf("\nValore di r non valido");
    return;
   }

 N=atoi(argv[3]);
 if (N<=0)
   {
    printf("\nValore di N non valido");
    return;
   }

 fd=fopen("points.lst","w");

 /*
    ciclo di iterazione ricorsiva della fun-
zione e di scrittura
    su file delle coppie {indice,valore-risul-
tato}
 */
 x=x0;
 for (i=0; i<=N; i++)
    {
     printf("\nX%i = %f",i,x);
     fprintf(fd,"{%i,%f}\n",i,x);
     x=x+r*x*(1-x/k);
    }

 fclose(fd);
}
```

E invece è proprio da qui che inizia il ... caos: è sufficiente variare (di poco!) il fattore di accrescimento - il tasso di crescita r - per ottenere comportamenti, cioè modalità di crescita della popolazione, del tutto imprevedibili. Il programma di simulazione *iter* permette di eseguire esperimenti virtuali: è infatti sufficiente ripeterne l'esecuzione variandone i parametri per osservare i cambiamenti indotti nella rappresentazione grafica dei risultati. Manteniamo per il momento costanti x_0 (0.2) e k (1.0) e proviamo a modificare r. Per valori di r minori di 2 il risultato è sostanzialmente quello illustrato nella figura 3: la variazione di r ha in questo caso come unica conseguenza la variazione della velocità con cui la popolazione si evolve, anche con piccole oscillazioni, verso la condizione di equilibrio stazionario. Ma per r= 2.2 (**figura 4**) anzichè uno stato di equilibrio la popolazione raggiunge velocemente una condizione di oscillazione periodica tra due valori.

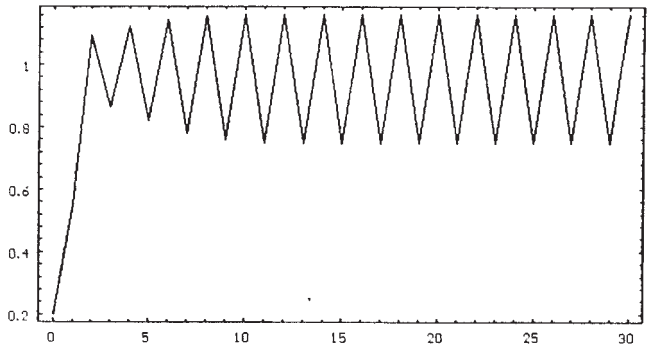


Figura 4 - Per r= 2.2 anziché uno stato di equilibrio, la popolazione raggiunge velocemente una condizione di oscillazione periodica tra due valori.

Questo inaspettato comportamento si mantiene per valori crescenti di r fino a 2.5: a partire da questo valore la popolazione, dopo una fase transitoria iniziale di rapida crescita e di oscillazione irregolare, si mantiene in una condizione di oscillazione periodica tra quattro valori (le **figure 5** e **6** sono relative al valore r= 2.52).

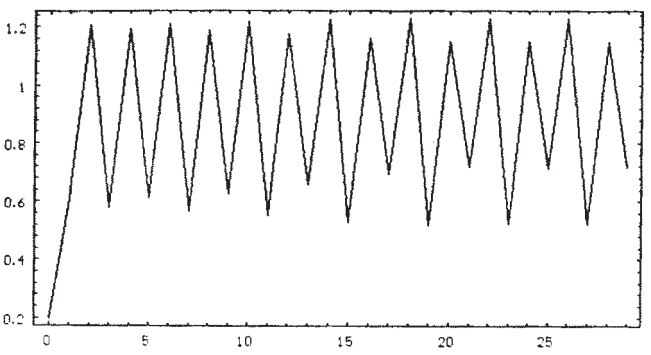


Figura 5 - Per r>2.5 la popolazione, dopo una fase transitoria iniziale di rapida crescita e oscillazione irregolare, si mantiene in una condizione di oscillazione periodica tra quattro valori.

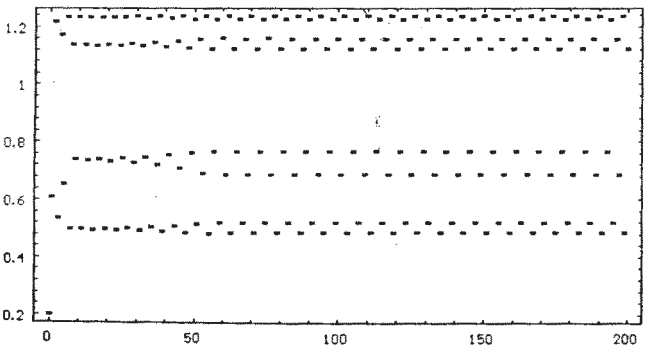


Figura 6 - Fino al valore r=2.57, per variazioni sempre più piccole del fattore di accrescimento si ottiene di volta in volta un raddoppiamento dei punti della fase di oscillazione periodica che segue il transitorio iniziale.

Fino al valore r = 2.57 per variazioni sempre più piccole del fattore di accrescimento si ottiene di volta in volta un raddoppio dei punti della fase di oscillazione periodica che segue il transitorio iniziale.
Da un punto di vista questa estensione del concetto di stato di equilibrio prende il nome di ciclo attrattore, mentre il valore di r per il quale si ha un raddoppio dei punti del ciclo viene denominato punto di biforcazione: pertanto si ottengono cicli attrattori formati da 2, 4 ,8, 16, 32, 64, ... si tratta di valori e punti di biforcazione sempre più vicini al precedente punto (la costante di Feigenbaum - che caratterizza molti modelli della teoria del caos - è calcolata a partire da una semplice relazione tra i punti di biforcazione successivi).
Come conseguenza si ha che anche una piccola incertezza nella stima del fattore di crescita r comporta l'impossibilità di prevedere l'evoluzione delle future generazioni della popolazione.
Per studiare il comportamento del modello per valori di r maggiori di 2.57 si dimostra utile la costruzione di un grafico che riporti - in funzione di r - i valori di oscillazione della popolazione dopo il transitorio iniziale. Una volta osservato che i valori della lista prodotta da *iter* rappresentano il risultato dell'applicazione ricorsiva della funzione $\mathbf{f(x)=-rx^2+(1+r)x}$ al valore x_0 - per esempio: $x_7=f(f(f(f(f(f(f(x_0)))))))$ - è possibile utilizzare direttamente il linguaggio di programmazione di *Mathematica* per costruire il grafico - conosciuto come diagramma di biforcazione e transizione al caos - riportato nella figura 7 (i comandi di *Mathematica* utilizzati per produrlo sono riportati, e brevemente commentati, nel **listato 2**).
Nel grafico sono chiaramente individuabili i cicli attrattori e i punti di biforcazione per valori del parametro r (rappresentato in ascissa) minori di 2.57; oltre questo valore critico si ha quello che viene definito "comportamento caotico": anche dopo la fase transitoria iniziale la popolazione si evolve con oscillazioni non periodiche e irregolari - imprevedibili - e con modalità fortemente dipendenti dai valori di r e di x_0 che rappresentano le condizioni iniziali.

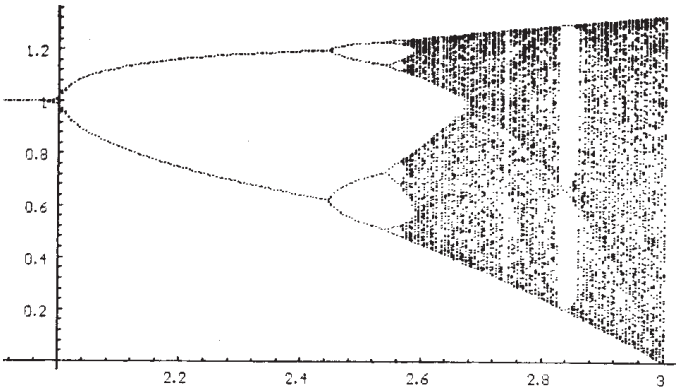


Figura 7 - Il diagramma di biforcazione e transazione al caos.

LISTATO 2

```
(* nella lista "list" si inseriscono le coor-
dinate dei punti del grafico *)
list = {};
(* il valore di r rappresenta la variabile
indipendente del grafico *)
For [r=1.9, r<3, r+=0.005,
   (* valore iniziale *)
   x=0.2;
   (* si trascurano i primi 100 valori dell'i-
terazione funzionale *)
   Do [x=-r*x^2+(1+r)*x, {i, 100}];
   (* nella lista "y" si inseriscono i 100
valori successivi *)
   y = {};
   Do [x=-r*x^2+(1+r)*x;
   (* si esegue un arrotondamento del risulta-
to alla terza cifra decimale *)
     v = N[Round[1000*x]/1000.0];
     (* si trascurano i valori ripetuti *)
     (* l'uso di "y" rende veloce il controllo
di pre-esistenza del valore *)
     If [MemberQ[y, {r, v}]==False,
       AppendTo [y, {r, v}];
       AppendTo [list, {r, v}];
       ],
     {i, 100}
   ];
];
(* rappresentazione grafica di "list" *)
ListPlot[list];
```

Le bande di comportamento caotico possono presentare un comportamento stocastico (in alcuni modelli il parametro di controllo prende il nome di parametro di stocasticità) e essere quindi studiate con modelli probabilistici e tecniche statistiche, ma possiedono in genere una elevata strutturazione deterministica: si noti ad esempio che esistono valori di r (compresi tra 2.8 e 2.85) dove compare nuovamente un ciclo attrattore formato da 3 valori, seguito da una banda di biforcazione con cicli attrattori di 6, 12, 24, 48, ... punti e da una nuova banda caotica (la figura a colori - realizzata con Fractint for Windows, freeware prodotto da The Stone Soup Group - rappresenta un ingrandimento orizzontale di questa parte del grafico).
La figura 8 rappresenta un ingrandimento orizzontale di un particolare della figura 7 (relativo ai valori di r compresi tra 2.5 e 2.6 ed ottenuto con i comandi del **listato 3**): non è difficile notare che l'intero diagramma di biforcazione vi è riprodotto in scala più piccola.
Questa proprietà di "autosomiglianza" a scale diverse è una caratteristica degli oggetti frattali, così descritti da Ian Malcom:

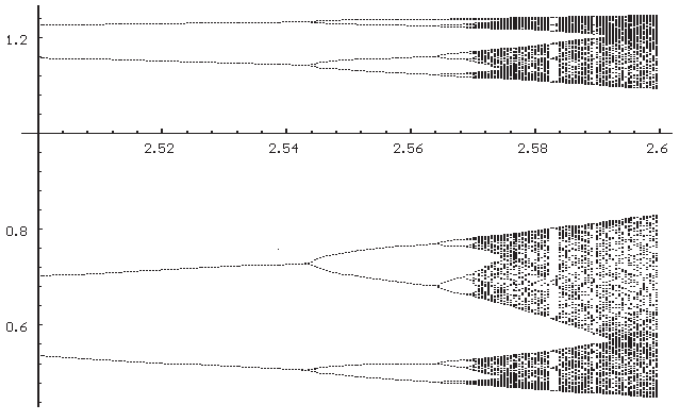


Figura 8 - Ingrandimento orizzontale di un particolare della figura 7 (per 2 ≤ 2.5 ≤ r ≤ 2.6).

LISTATO 3

```
list = {};
(* modificato il campo dei valori di r e il
passo di scansione *)
For [r=2.5, r<2.6, r+=0.0005,
   x=0.2;
   (* modificato il numero dei valori scartati
come transitori *)
   Do [x=-r*x^2+(1+r)*x, {i, 500}];
   y = {};
   Do [x=-r*x^2+(1+r)*x;
     v = N[Round[1000*x]/1000.0];
     If [MemberQ[y, {r, v}]==False,
       AppendTo [y, {r, v}];
       AppendTo [list, {r, v}];
       ],
     (* modificato il numero dei valori presi
in considerazione *)
     {i, 500}
   ];
];
ListPlot[list];
```

"- I frattali sono una specie di geometria sviluppata in particolare da un certo Mandelbrot, ma, a differenza di quella euclidea che tutti imparano a scuola - quadrati, cubi e sfere - la geometria frattale sembra descrivere oggetti reali del mondo naturale. Monti e nuvole sono forme frattali. E i frattali sono probabilmente in rapporto con la realtà. In qualche modo.
- Be', Mandelbrot, coi suoi strumenti geometrici, ha fatto un'interessante scoperta. Ha trovato che le cose sembrano quasi identiche in scale differenti -.

[...]
- Per esempio -, disse Malcom, - una grande montagna, vista da lontano, ha una sua forma montagnosa ed aspra. Andando più vicino ed esaminando un piccolo picco della grande montagna, ritroveremo quella stessa forma. E scendendo via via sino a un frammento di roccia visto al microscopio, ritroveremo sempre la forma frattale di base della grande montagna -.
[Michael Crichton, Jurassic Park, III ed. it., Garzanti, 1993, pag. 216]

Molti modelli della teoria del caos presentano una "geometria" frattale: la fusione tra queste due discipline matematiche relativamente nuove e fondate sulle potenzialità di calcolo numerico e grafico rese disponibili dai moderni calcolatori costituisce uno strumento concettuale di indagine della realtà che trova sempre maggiori conferme da parte di ricerche condotte nei campi più diversi, ma che presenta al contempo molti punti di rottura rispetto al metodo scientifico "classico". Si abbandona in primo luogo la pretesa di prevedere quantitativamente un evento (di quanti individui sarà composta la popolazione tra 100 generazioni?) per intraprendere uno studio qualitativo dei processi (che tipo di oscillazione presenterà la popolazione nelle generazioni future?): è un approccio conoscitivo - un paradigma secondo i filosofi della scienza - completamente nuovo, con importanti, ma non riconosciute, conseguenze nel campo della tecnologia, dell'ecologia e dell'economia (nell'articolo citato, May ricordava che non dovrebbe essere indifferente per i responsabili politici sapere che piccoli interventi locali possono avere vasti e imprevedibili effetti globali). Ma riconoscere che la quasi totalità dei sistemi reali non può essere descritta con un modello lineare rende inservibile ai fini conoscitivi l'approccio riduzionista che ha caratterizzato fino ad oggi l'indagine scientifica e secondo il quale lo studio di un sistema complesso viene ridotto allo studio indipendente delle sue parti componenti: se le interazioni tra le componenti non sono lineari, la conoscenza del loro comportamento non permette di comprendere il comportamento del sistema globale. Dall'evidenza di questa severa limitazione è nata la discussa "teoria della complessità", ma questa è un'altra storia. Ascoltiamo invece le conclusioni in proposito di Ian Malcom:

"- [...] la scienza ha perso la sua grande giustificazione intellettuale. Fin dai tempi di Newton e Cartesio, la scienza ci ha offerto esplicitamente la prospettiva di un controllo totale. La scienza ha sostenuto di potere effettivamente controllare ogni cosa grazie alla sua comprensione delle leggi naturali. Ma nel ventesimo secolo questa pretesa è franata inesorabilmente. [...]. Poi il teorema di Godel ha posto limiti simili alla matematica, il linguaggio formale della scienza. Una volta i matematici pensavano che al loro linguaggio fosse connaturata una speciale verità intrinseca derivata dalle leggi della logica. Ora sappiamo che quanto chiamiamo "ragione" non è che un gioco arbitrario. Non è speciale, non lo è come pensavamo.
- E adesso la teoria del caos prova che l'imprevedibilità è insita nella nostra vita quotidiana. E` altrettanto comune quanto la tempesta che non possiamo predire. E così la grande prospettiva della scienza, vecchia centinaia d'anni! - il sogno di un controllo totale - è morta, nel nostro secolo. E con essa molte sue giustificazioni, la ragione della scienza di fare quanto fa. Come pure la nostra ragione di ascoltarla. La scienza ha sempre sostenuto che, pur non sapendo tutto in un dato momento, sarà in grado di scoprirlo più tardi. Ma adesso ci siamo resi conto che questo non è vero. E` una vuota millanteria. Altrettanto sciocca e poco giudiziosa quanto il bimbo che salta giù dal tetto di una casa convinto di poter volare-. [...].
- Che cosa accadrà allora? -, chiese Ellie.
Malcom si strinse nelle spalle. - Un cambiamento -.
- Che genere di cambiamento ? -.
- Tutti i più grandi cambiamenti sono come la morte -, disse.
- Non puoi vedere l'altro lato finché non sei di là -. E chiuse gli occhi."
[Michael Crichton, Jurassic Park, III ed. it., Garzanti, 1993, pagg. 379-380]

Al termine del libro - nei ringraziamenti - Michael Crichton scrive che la figura di Ian Malcom è ispirata alle "opere del compianto Heinz Pagels".

Bibliografia ragionata

[1] Michael Crichton, *Jurassic Park*, trad. it. di Maria Teresa Marenco e Andrea Pagnes, Garzanti, 1993 (III ed.)

A chi vuole approfondire in modo rigoroso, ma non specialistico, la portata del "testamento filosofico" di Ian Malcom consiglio il contributo di Marco d'Eramo - *L'abisso non sbadiglia più* - ad una interessante raccolta di articoli sul tema della teoria del caos, ma con sconfinamenti in quasi tutti i campi "critici della scienza contemporanea, scritti in un linguaggio senz'altro accessibile:
[2] Aa. Vv., Gli ordini del caos, ed. manifestolibri, 1991

Un famoso libro divulgativo sulla teoria del caos è:
[3] James Gleick, *Il caos*, trad. it., Rizzoli, 1989

Per chi vuole conoscere i temi e le applicazioni della teoria del caos segnalo la seguente raccolta di articoli che comprende *Attrattori strani: enti fra ordine e caos* di Douglas R. Hofstadter (l'autore del famoso *Godel, Escher, Bach*):
[4] Aa. Vv., Il Caos: le leggi del disordine, Le Scienze ed., 1991

I comandi che ho utilizzato per generare con *Mathematica*

i grafici di biforcazione e transizione al regime caotico si trovano nel paragrafo 6.3 (“Chaos and fractals”) di:
[5] Richard Crandall, *Mathematica* for the Sciences, Addison-Wesley, 1991.

Stephen Wolfram, oltre ad essere il principale autore di *Mathematica*, è fisico, matematico ed informatico di riconosciuto valore e scienziato nel campo della teoria dei sistemi complessi e delle dinamiche non lineari; questo suo articolo divulgativo, pur parlando principalmente degli “automi cellulari” - un altro argomento di grande interesse interdisciplinare - focalizza l’importanza del computer nella ricerca teorica ed applicata contemporanea:
[6] Stephen Wolfram, Software nella scienza e nella matematica, Le Scienze (ed. it. di *Scientific American*) n. 159, Novembre 1984.

Questo bellissimo libro esplora gli arcani della Matematica servendosi di *Mathematica*, è scritto in forma di dialogo tra un saggio insegnante - Theo - ed un alunno curioso - Jerry - ed ha un capitolo - il settimo: “Bifurcations Forever” - dedicato allo studio approfondito delle biforcazioni derivanti dall’iterazione ricorsiva della funzione quadratica:
[7] Theodore Gray & Jerry Glynn, Exploring Mathematics with *Mathematica:* Dialogs Concerning Computers and Mathematics, Addison-Wesley, 1991.

0. Lo "studio del comportamento caotico e frattale del calcolo iterato di una funzione quadratica" apre alcune prospettive di approfondimento: da una parte la geometria degli oggetti frattali aventi caratteristiche di auto-somiglianza ed invarianza di scala, dall'altra l'analisi dei modelli di variazione della consistenza numerica delle popolazioni nel caso di due specie in competizione (sistemi preda-predatore).

1. La formalizzazione del concetto di dimensione geometrica frazionaria di una figura frattale risulta essere troppo complessa per essere proposta nella scuola media superiore, ma la lettura di alcune provocazioni didattiche di Mandelbrot ("Quanto è lunga la costa della Gran Bretagna?") è, almeno per quanto riguarda la parte divulgativa, senz'altro proponibile agli studenti. Sembrerebbe naturale il passaggio, nel calcolo iterato di una funzione quadratica, dal campo reale a quello complesso e quindi al famoso "insieme di Mandelbrot" ed alle sue variopinte rappresentazioni grafiche, dove i colori codificano la velocità di convergenza dell'algoritmo numerico di calcolo. Riteniamo più significativo affrontare lo studio delle forme frattali a partire da alcuni strumenti matematici di cui lo studente già dispone, per rivelarne nuove potenzialità: le trasformazioni affini del piano cartesiano e le grammatiche generative in congiunzione con la geometria procedurale della tartaruga del linguaggio Logo, comunemente introdotta nella scuola media inferiore.

    1.1. Un "insieme iterato di funzioni" (IFS: Iterated Function System) è definito da una figura geometrica iniziale (generalmente un poligono individuato per mezzo delle coordinate cartesiane dei propri vertici) e da un elenco di trasformazioni affini e contrattive del piano caratterizzate dalle rispettive equazioni: applicando ripetutamente (ma, ovviamente, interrompendo la ricorsione ad un livello predefinito di profondità) le trasformazioni alla figura iniziale, si ottiene una rappresentazione grafica dell'insieme frattale descritto dallo specifico IFS. Un risultato fondamentale relativo ai sistemi iterati di funzioni è che la struttura frattale finale è completamente indipendente dalla figura iniziale -che non influenza le approssimazioni di ordine più elevato- ed è esclusivamente individuata dalle trasformazioni geometriche che a questa si applicano! È questo un caso particolare del teorema del punto fisso.

    1.2. I sistemi-L (Lindenmayer) sono grammatiche formali comprendenti un numero limitato di regole di "riscrittura" che, applicate ricorsivamente, generano forme frattali regolari, ma potenzialmente ricche di infiniti livelli di dettaglio. Le stringhe generate dalla grammatica di un sistema-L sono interpretate graficamente come comandi di una geometria procedurale molto simile a quella della tartaruga del linguaggio Logo. I sistemi-L si differenziano dalle grammatiche libere dal contesto perché non permettono l'esistenza di una molteplicità di produzioni che trasformino in

stringhe diverse lo stesso simbolo non terminale; in questo modo a partire da una stringa iniziale denominata "assioma", e non da un singolo simbolo come nel caso delle grammatiche, non si definisce un insieme potenzialmente infinito di stringhe (il linguaggio), ma si ottiene un'unica stringa derivata: in realtà i sistemi-L implementano un "sistema di sostituzioni". Inoltre la distinzione tra simboli terminali e non terminali non è essenziale in quanto l'applicazione ricorsiva delle regole di trasformazione alle stringhe di simboli non termina con l'assenza di simboli non terminali, come nel caso delle grammatiche generative, ma dopo un numero di passi specificato a priori (ad ogni passo vengono effettuate tutte le sostituzioni rese possibili dall'applicazioni delle regole di trasformazione).

2. Il problema dell'analisi di un sistema predatore-preda può essere affrontato con la realizzazione di un programma di simulazione o per mezzo dello studio dei classici sistemi di equazioni differenziali di Lotka-Volterra, facilmente integrabili numericamente ricorrendo alle funzionalità di Mathematica o di Maple. Questi stessi programmi permettono di rappresentarne graficamente le soluzioni rispetto al trascorrere del tempo o nello "spazio delle fasi", anche in forma parametrica. Interessante è il fatto che la simulazione, che comunque è una grossolana approssimazione della realtà, produce risultati qualitativamente in accordo con il modello matematico differenziale, ma quantitativamente imprevedibili a partire da questo: infatti se rappresentati su uno spazio delle fasi ricordano molto un'attrattore caotico e frattale anziché il ciclo o il punto limite previsti dal modello.

frattale, *agg.* Senso intuitivo: di forma estremamente irregolare, o estremamente interrotta e frammentata, e che rimane tale qualunque sia la scala a cui la si esamina; contenente degli elementi distintivi le cui scale sono molto varie e coprono una gamma molto larga. Motivazione: i matematici si erano occupati, da cento anni a questa parte, di alcuni degli insiemi in questione, ma non avevano costruito nessuna teoria al riguardo e non avevano perciò avvertito la necessità di un vocabolo che li designasse. Da quando l'autore ha mostrato che la natura rigurgita di oggetti le cui migliori rappresentazioni matematiche sono fornite da insiemi frattali si è reso necessario un termine appropriato e dal significato univoco. DIMENSIONE FRATTALE. 1. Senso generico: numero che serve a quantificare il grado di irregolarità e di frammentazione di un insieme geometrico o di un oggetto naturale; la dimensione frattale non è necessariamente un intero. 2. Senso specifico: dimensione nel senso di Hausdorff e Besicovitch. INSIEME FRATTALE. Insieme di dimensione frattale superiore alla sua dimensione ordinaria (topologica). OGGETTO FRATTALE. Oggetto naturale che è ragionevole e utile rappresentare matematicamente a mezzo di un insieme frattale. **frattale**, *s.m.* Configurazione frattale; insieme o oggetto frattale. Osservazione: *frattale* non distingue, di proposito, tra insiemi matematici (teoria) e oggetti naturali (realtà): si impiega in tutti i casi in cui la sua generalità, e la deliberata ambiguità che ne risulta, siano desiderabili, prive di inconvenienti, e sollecitate dal contesto.

[B. Mandelbrot, *Lessico dei neologismi*, appendice a "Gli oggetti frattali: forma, caso e dimensione", ed. it. a cura di R. Pignoni, Einaudi, 1987]

Creazione di figure frattali biomorfe

Giorgio Meini

La prima edizione di *The Fractal Geometry of Nature* **[1]** di Benoit Mandelbrot del Thomas Watson Research Center della IBM è stata pubblicata nel 1982; da allora l'idea di una descrizione matematica della natura (coste, montagne, galassie, piante, ...) non più fondata sulla classica geometria di "rappresentazione" delle forme, ma sulla geometria "generativa" dei frattali, ha incontrato numerose conferme e notevole successo. Come conseguenza della caratteristica proprietà di "invarianza di scala" e contrariamente alle forme della geometria classica una conformazione frattale può essere rappresentata esplicitamente (graficamente) solo con risoluzione approssimata: una compiuta descrizione dei dettagli di un frattale è invece implicitamente contenuta nel procedimento algoritmico e ricorsivo di generazione.

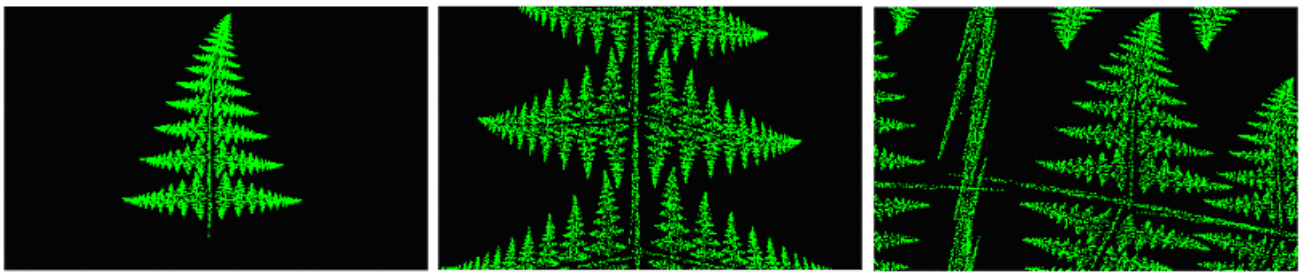


Figura 1

Nella **figura 1** sono rappresentati dettagli autosimili presenti a scale di osservazione diverse di un famoso frattale biomorfo, la felce. Sempre di Mandelbrot è l'idea suggestiva che le forme di molti

oggetti naturali siano generate da perturbazioni casuali, da interpretarsi come sintesi delle interazioni che l'oggetto ha con l'ambiente, di un frattale altrimenti deterministico:

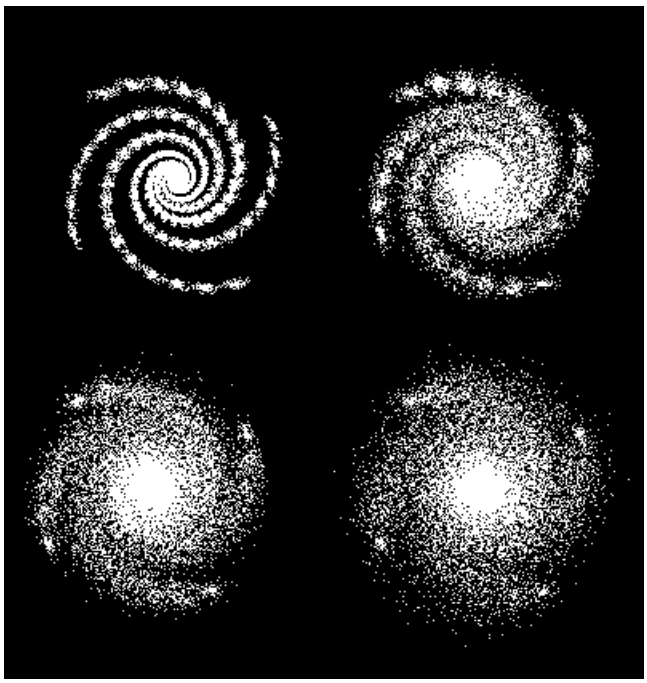


Figura 2

nella **figura 2** si hanno quattro "galassie" ottenute impiegando lo stesso algoritmo di costruzione, ma con coefficiente di casualità posto rispettivamente uguale a 0.1, 0.3, 0.5 e 0.8. L'innegabile realismo di queste immagini non deve però trarre in inganno: il problema della relazione che sussiste tra la realtà e la sua descrizione matematica è complesso e l'equivalenza di forme non garantisce di per sé una effettiva corrispondenza tra il procedimento di costruzione del frattale e il processo generativo naturale. In ogni caso un importante contributo derivante dall'applicazione della geometria frattale allo studio delle scienze naturali consiste nello spostamento di attenzione dal prodotto finito - la forma statica e definitiva - al processo dinamico di generazione della forma stessa. Il programmatore interessato a realizzare in proprio un semplice programma che applica la tecnica della perturbazione casuale alla generazione di un frattale che descrive il profilo di una montagna troverà suggerimenti essenziali nella rubrica "(Ri)creazioni al calcolatore" del numero 222 della rivista "Le Scienze" **[2]**. La ricorsività intrinseca degli algortimi di generazione dei frattali è la causa della tipica proprietà di autosomiglianza a scale di osservazione diverse delle loro rappresentazioni grafiche ed è proprio questa caratteristica che permette alle forme frattali di essere adeguati modelli almeno di alcune delle forme di vita meno complesse: questa analogia è limitata alla sola conformazione esteriore?

Nel 1990 Prusinkiewicz e Lindenmayer pubblicano *The Algorthmic Beauty of Plants* in cui si prende in considerazione l'ipotesi di un programma genetico che, applicando ricorsivamente le "regole" dello sviluppo proprie della specie, generi una morfologia frattale; ancora una volta la varietà morfologica individuale viene simulata con perturbazioni casuali sostitutive delle interazioni ambientali particolari. Il modello di sviluppo proposto è oggi noto con il nome di *L-systems*: si tratta di una grammatica formale comprendente un numero limitato di regole di "riscrittura" che applicate ricorsivamente generano *pattern* frattali regolari, ma potenzialmente ricchi di infiniti livelli di dettaglio.

Le grammatiche formali

Una grammatica formale (libera dal contesto) consiste essenzialmente in un insieme di regole di trasformazione (sostituzioni) di stringhe di simboli (generalmente caratteri alfabetici). Queste regole di riscrittura sono denominate "produzioni" perché permettono, a partire da una data stringa di simboli, di produrne una nuova. La definizione formale di una grammatica comprende inoltre: un alfabeto di simboli "terminali" che non possono essere sostituiti dall'applicazione delle produzioni, un elenco di simboli "non terminali" (categorie o domini sintattici) oggetto di trasformazione da parte delle produzioni e un simbolo iniziale scelto tra i non terminali. Tutte le produzioni hanno la forma S $\rightarrow$ s, dove *S* è un singolo simbolo non terminale e *s* è una stringa di simboli (terminali e non terminali) sostituibile ad *S*. Una grammatica formale definisce un insieme di stringhe sintatticamente corrette (linguaggio) derivabili a partire dal simbolo iniziale per mezzo della ripetuta applicazione (non deterministica) delle regole. La seguente grammatica definisce ad esempio un sotto-insieme (infinito) delle espressioni sintatticamente corrette di un linguaggio di programmazione:

simboli terminali: x,y,z,+,*,(,);
simboli non terminali: E;
simbolo iniziale: E;
produzioni: (a) E$\rightarrow$x, (b) E$\rightarrow$y, (c) E$\rightarrow$z, (d) E$\rightarrow$(E), (e) E$\rightarrow$E+E, (f) E$\rightarrow$E*E.

La stringa "(x+y)*z" rappresenta una espressione corretta (appartiene cioè al linguaggio generato dalla grammatica) perché può essere derivata in 6 passi applicando ricorsivamente le produzioni al simbolo iniziale:

1) produzione *f*: E $\rightarrow$ E*E,
2) produzione *d*: E*E $\rightarrow$ (E)*E,
3) produzione *c*: (E)*E $\rightarrow$ (E)*z,
4) produzione *e*: (E)*z $\rightarrow$ (E+E)*z,
5) produzione *a*: (E+E)*z $\rightarrow$ (x+E)*z,
6) produzione *b*: (x+E)*z $\rightarrow$ (x+y)*z.

Gli *L-systems* si differenziano dalle grammatiche libere dal contesto perché non permettono l'esistenza di una molteplicità di produzioni che trasformino in stringhe diverse lo stesso simbolo non terminale; in questo modo a partire da una stringa iniziale denominata assioma - e non da un singolo simbolo come nel caso delle grammatiche - non si definisce un insieme potenzialmente infinito di stringhe (il linguaggio), ma si ottiene un'unica stringa derivata: in realtà gli L-system implementano un "sistema di sostituzioni". Inoltre la distinzione tra simboli terminali e non terminali è inessenziale in quanto l'applicazione ricorsiva delle regole di trasformazione alle stringhe di simboli non termina con l'assenza di simboli non terminali come nel caso delle grammatiche generative, ma dopo un numero di passi (ad ogni passo vengono operate tutte le sostituzioni rese possibili dall'applicazione delle regole di trasformazione) specificato a priori.

Gli L-system di Lindenmayer sono stati implementati nel diffuso software *freeware* **Fractint** da Adrian Mariano a partire da una geometria procedurale molto simile a quella della "tartaruga" del linguaggio Logo (chi ne è completamente all'oscuro legga "Guardando la geometria dal di dentro, a passeggio con una tartaruga" di Brian Hayes, nel numero 188 dell'Aprile 1984 della rivista "Le Scienze"). La geometria della tartaruga permette di disegnare segmenti per mezzo dei seguenti comandi:

F traccia un segmento di lunghezza unitaria,
G spostati di un segmento di lunghezza unitaria (senza disegnarlo),
+ modifica la direzione di un angolo unitario in senso orario,
- modifica la direzione di un angolo unitario in senso antiorario,
| modifica la direzione di un angolo di 180° (se possibile),
! inverti la direzione,
@*n* modifica la dimensione del segmento unitario moltiplicandola per il coefficiente *n*,

[push: memorizza nello stack posizione, direzione e dimensione segmento unitario,
] pop: richiama dallo stack posizione, direzione e dimensione segmento unitario.

Un file .L per Fractint (si veda il **listato 1** per un esempio) è costituito da più sistemi-L ciascuno dei quali è contraddistinto da un nome e composto da una specificazione dell'ampiezza dell'angolo unitario in frazioni dell'angolo giro (*angle 4* per un angolo di 90°), dall'assioma iniziale (*axiom*) e dall'elenco delle regole di riscrittura da applicare all'assioma per un numero di volte da specificarsi al momento della rappresentazione. Prendiamo in esame un primo e semplice esempio di sistema L:

```
KochCurve { ; costa dell'isola di Koch (1904)
  Angle 6
  Axiom F
  F=F+F--F+F
  }
```

La regola di riscrittura stabilisce che ogni carattere F debba essere sostituito con la stringa F+F--F+F (l'angolo unitario è posto pari a 360°/6=60°) e viene ricorsivamente applicata a partire dall'assioma iniziale (F) un numero di volte (denominato ordine di applicazione o livello di profondità) specificato a priori; osserviamo le trasformazioni del risultato al variare del livello di applicazione:

0 F (assioma),
1 F+F--F+F,
2 F+F--F+F + F+F--F+F -- F+F--F+F + F+F--F+F,
3 F+F--F+F + F+F--F+F -- F+F--F+F + F+F--F+F + F+F--F+F + F+F--F+F -- ...;

al termine delle sostituzioni successive Fractint interpreta la stringa ottenuta come sequenza di comandi grafici (la **figura 3** rappresenta le rappresentazioni approssimate di ordine 1, 2, 3 e 4).

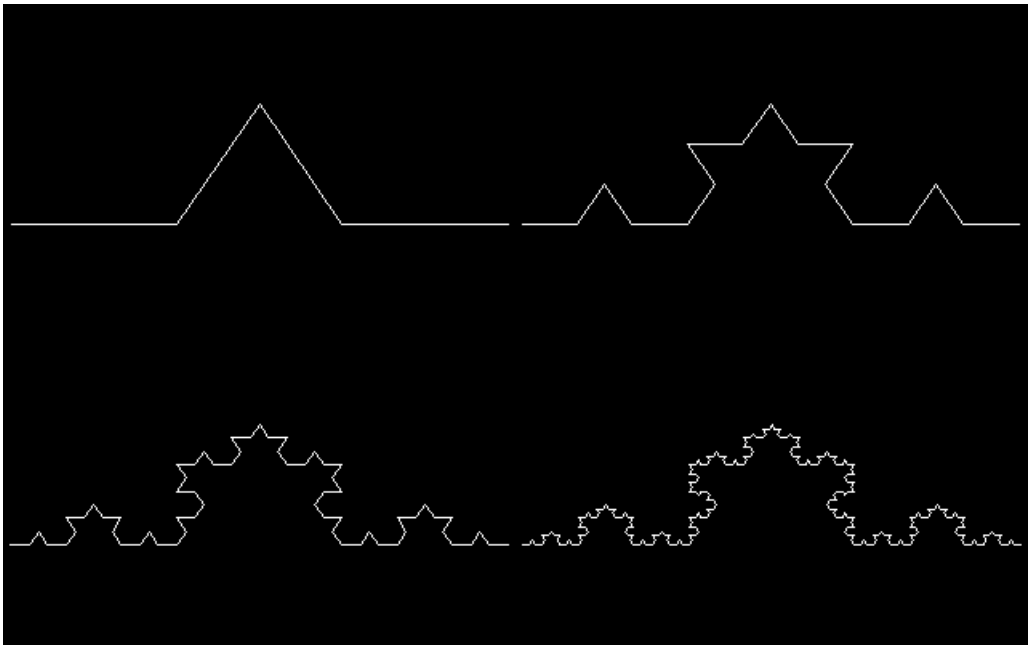


Figura 3

Nonostante l'apparente laboriosità il metodo risulta essere estremamente adatto a descrivere oggetti frattali composti da segmenti e con strutture autosomiglianti a scale di osservazione diverse; è possibile inserire nell'assioma ed utilizzare per costruire le regole quasi tutti i caratteri disponibili: terminate le sostituzioni i caratteri non corrispondenti a comandi grafici verranno semplicemente ignorati. L'esempio della curva di Koch è però molto semplice; per ottenere strutture simili alle piante reali (si vedano ad esempio le **figure 4 e 5** ed i relativi L-system generativi nel **listato 1**)

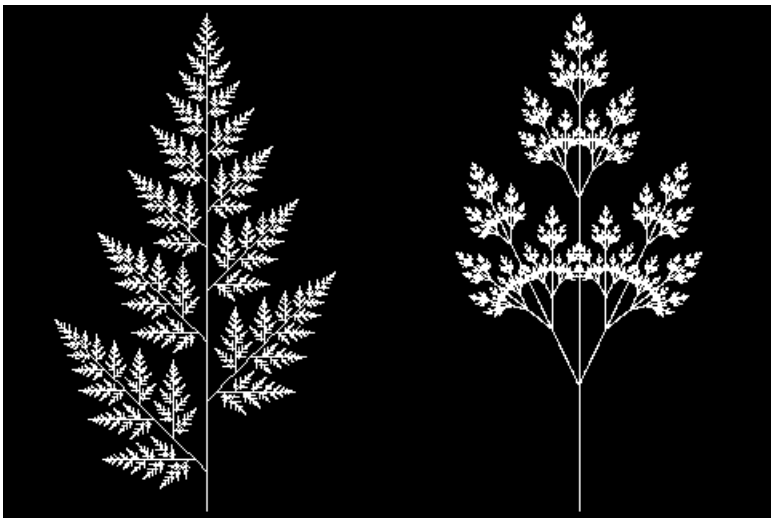


Figure 4 e 5

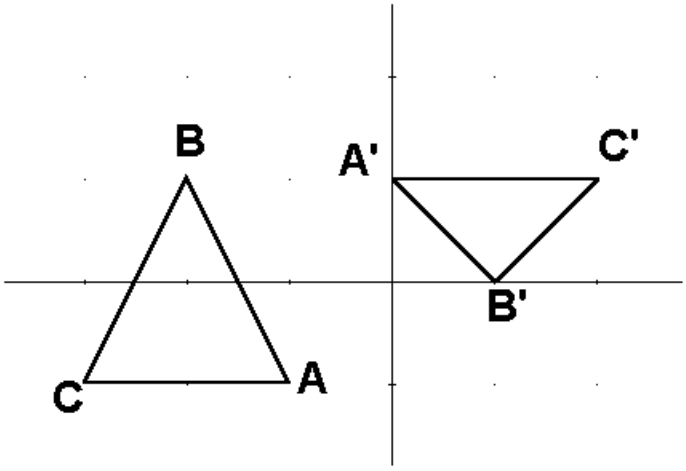
si ricorre a sistemi di regole più complesse e mutuamente ricorsive. Si noti che l'innaturalità delle piante raffigurate in queste immagini è dovuta soprattutto alla loro simmetria ed invarianza di scala assolutamente perfetta; per ottenere immagini più realistiche è sufficiente introdurre una limitata perturbazione casuale nel procedimento di generazione: invito tutti i lettori a collaborare realizzando un programma *ad hoc*. Un programma di rappresentazione di frattali non deterministici generati da L-system (rigidamente deterministici!) non può semplicemente esibire un comportamento come quello preso in esame a proposito delle galassie della figura 2 (dispersione casuale di punti in un "intorno" - di ampiezza specificata da un coefficiente di stocasticità - del *pattern* generato): è necessario introdurre perturbazioni casuali della "struttura" descritta dalle regole del sistema. È possibile perturbare la struttura agendo sulle dimensioni dei segmenti, sulle ampiezze degli angoli e, se si ha cura di mantenere la proprietà di "connessione", sulla stessa configurazione descritta dalle regole del sistema; la perturbazione può precedere l'applicazione ricorsiva delle regole all'assioma ed essere quindi staticamente replicata in tutte le parti della rappresentazione (in questo modo si ottengono variazioni individuali, ma non si accresce il realismo dell'oggetto generato), oppure prodursi dinamicamente in una qualsiasi delle fasi successive di applicazione delle regole: in questo caso la "mutazione" può essere localizzata ad un solo livello o propagarsi per "eredità" ai livelli successivi. Buon lavoro!
I sistemi-L interpretano due evidenti caratteristiche dei sistemi viventi: lo sviluppo globale è una composizione parallela di una logica operativa esclusivamente locale (la sequenzializzazione del processo di accrescimento è infatti una scelta implementativa indipendente dal parallelismo potenziale dei sistemi-L) ed esso viene realizzato tramite la ripetuta applicazione di regole "genetiche" ricorsive (con conseguente economia della codifica). Come modello biologico però gli L-system sono una sorta di riproposizione delle tesi preformistiche del secolo scorso che sostenevano la presenza di tutte le informazioni necessarie alla formazione di un individuo completamente sviluppato già in fase embrionale. L'acquisita conoscenza dell'esistenza di un "programma" genetico comune a tutti gli organismi viventi - di cui il DNA è il supporto biochimico - orienta spesso verso una visione simile a questa dei processi biologici di sviluppo, ma in realtà lo

sviluppo di un individuo è il risultato "epigenetico" dell'interazione reciproca tra gli effetti del suo programma genetico e l'ambiente nel quale avviene il processo stesso: le proprietà del prodotto finale non sono contenute in un codice centralizzato, ma vengono ri-create ogni volta! Inoltre, nonostante che gli L-system forniscano una soddisfacente descrizione della struttura globale finale (almeno nel caso di alcune piante), non sono in grado di modellare la varietà dei particolari che costituiscono l'intero: un ago di pino è infatti strutturalmente diverso dai rami e dal tronco dello stesso albero!

Nel 1988 viene pubblicato *Fractals Everywhere* **[3]** di Michael Barnsley: è la data di nascita dei "sistemi iterati di funzioni" (*Iterated Function Systems* o IFS), una tecnica fondata sulle trasformazioni affini contrattive del piano geometrico per la codifica di insiemi frattali lineari e deterministici.

Le trasformazioni geometriche del piano cartesiano

Fissando un riferimento cartesiano (sistema di coordinate numeriche) nel piano geometrico ogni punto viene ad essere univocamente individuato da una coppia di valori reali (x,y) ed una figura poligonale può essere propriamente rappresentata dalla lista delle coppie di coordinate corrispondenti ai punti di vertice. Ad ogni trasformazione geometrica del piano corrisponde un sistema di equazioni che permettono di calcolare, a partire dalle coordinate numeriche dei punti originali (per esempio i vertici di un poligono), le coordinate dei punti trasformati riferite al medesimo sistema cartesiano.



La figura illustra la trasformazione del triangolo di vertici A(-1,-1), B(-2,1) e C(-3,-1) nel triangolo di vertici A'(0,1), B'(1,0) e C'(2,1); come si può facilmente verificare le equazioni di questa trasformazione sono le seguenti:

$$X' = -X - 1$$
$$Y' = -0.5*Y + 0.5$$

Restringendo l'interesse alla classe delle trasformazioni geometriche descritte da equazioni lineari (affinità) la forma algebrica del sistema che descrive la trasformazione sarà la seguente:

X' = a*X + b*Y +e
Y' = c*X + d*Y +f

si noti che nell'esempio precedente: a = –1, b = 0, c = 0, d = –0.5, e = –1, f = 0.5.
Le affinità che compaiono nei sistemi iterati di funzioni di Barnsley sono affinità contrattive, trasformazioni cioè in cui ogni poligono trasformato è ridotto di un fattore di scala rispetto all'originale; è questa infatti la proprietà che garantisce l'esistenza di un punto fisso per la trasformazione.

A partire da questa particolare codifica per le rappresentazioni grafiche degli oggetti frattali è stata in seguito sviluppata la tecnica della compressione pseudo-frattale delle immagini (si veda l'articolo di Stefano Giordano, "La codifica frattale delle immagini", su Computer Programming n. 32, Gennaio 1995). Un IFS è definito da una figura geometrica iniziale (generalmente un poligono individuato per mezzo delle coordinate cartesiane dei propri vertici) e da un elenco di trasformazioni affini e contrattive del piano caratterizzate dalle rispettive equazioni: applicando ripetutamente (ma, ovviamente, interrompendo la ricorsione ad un livello predefinito di profondità) le trasformazioni alla figura iniziale, si ottiene una rappresentazione grafica dell'insieme frattale descritto dallo specifico IFS.

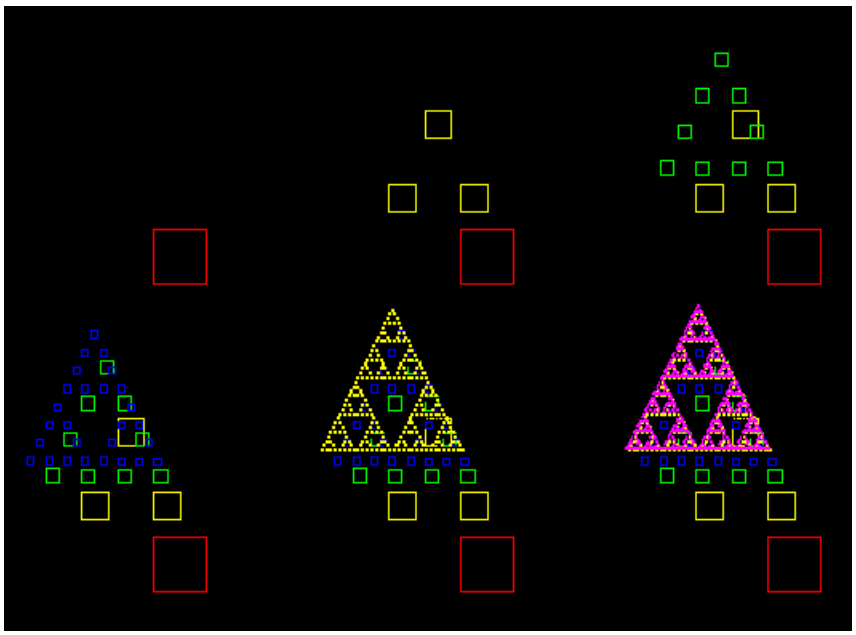


Figura 6

Nella **figura 6** sono rappresentati i primi livelli di profondità di un IFS che definisce la struttura del triangolo di Sierpinski (la prima immagine è la figura iniziale, la seconda corrisponde alla prima applicazione della trasformazione affine e così via ...). Le tre trasformazioni (corrispondenti alle tre diverse figure che si ottengono da ogni figura calcolata nell'iterazione immediatamente precedente) che generano il triangolo di Sierpinski hanno le seguenti equazioni:

X' = 0.5*X, Y' = 0.5*Y
X' = 0.5*X + 0.5, Y' = 0.5*Y
X' = 0.5*X + 0.25, Y' = 0.5*Y + 0.5

Riferendosi alle equazioni generali delle trasformazioni affini (X' = a*X + b*Y + e, Y' = c*X + d*Y + f) i parametri sono i seguenti:

a	b	c	d	e	f
0.5	**0**	**0**	**0.5**	**0**	**0**
0.5	**0**	**0**	**0.5**	**0.5**	**0**
0.5	**0**	**0**	**0.5**	**0.25**	**0.5**

Questa è anche la forma in cui gli IFS sono effettivamente rappresentati nel programma Fractint e il triangolo di Sierpinski può quindi essere così definito in un file .ifs (il significato dell'ultima colonna di valori verrà chiarito nel seguito):

```
Sierpinski {
 0.50     0.00     0.00     0.50     0.00     0.00     0.33
 0.50     0.00     0.00     0.50     0.50     0.00     0.33
 0.50     0.00     0.00     0.50     0.25     0.50     0.33
}
```

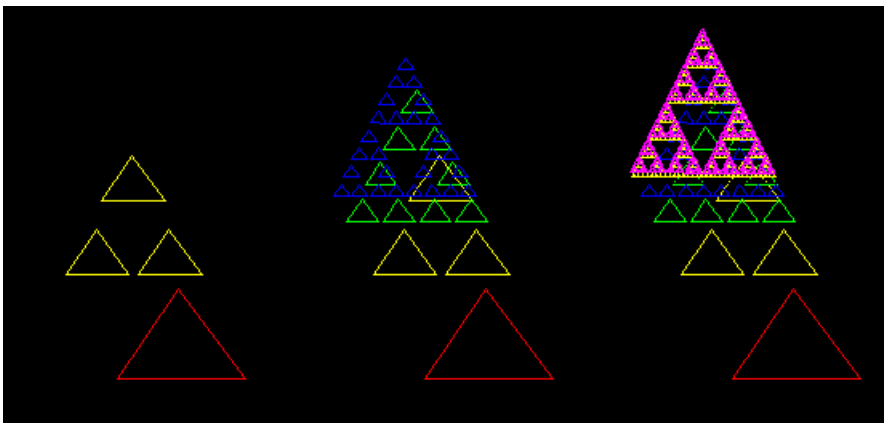


Figura 7

La **figura 7** - ancora relativa al triangolo frattale di Sierpinski - illustra un risultato fondamentale relativo ai sistemi iterati di funzioni: la struttura frattale finale è completamente indipendente dalla figura iniziale - che non influenza le approssimazioni di ordine più elevato - ed è esclusivamente individuata dalle trasformazioni geometriche che a questa si applicano! È questo un caso particolare del teorema del punto fisso: per una data classe di funzioni il valore limite che si ottiene dalla ripetuta applicazione della funzione stessa al valore calcolato nell'iterazione immediatamente precedente è indipendente dal valore iniziale; qualora lo stesso valore limite sia utilizzato come argomento della funzione lo si ottiene come risultato. Ad esempio nell'insieme dei numeri reali positivi la funzione $f(x)=e^{-x}$ ha il numero 0.567... come punto fisso: infatti f(0,567...)=0,567... e, scegliendo un qualsiasi numero reale positivo (per esempio x=0.5) come valore iniziale, l'applicazione iterata della funzione converge al valore limite 0.567..., come si può facilmente verificare per mezzo di una calcolatrice:

f(0.5)=0.607...
f(0.607)=0.545...
f(0.545)=0.580...
f(0.580...)=0.560...
f(0.560...)=0.571...
f(0.571...)=0.565...
f(0.565...)=0.568...

f(0.568...)=0.566...
f(0.566...)=0.567...
f(0.567...)=0.567...

Le **figure 6 e 7** (così come la **figura 1**) rappresentano sequenze di "accrescimento" di frattali descritti da sistemi iterati di funzioni: la configurazione finale è data dal limite a cui tende il procedimento di applicazione ricorsiva delle funzioni di trasformazione del sistema e costituisce un attrattore indipendente dalla configurazione iniziale. Se sono necessarie *n* iterazioni per rendere trascurabile il contributo della configurazione iniziale alla costruzione approssimata della configurazione limite e se il sistema di funzioni è composto da *N* equazioni, allora si devono calcolare $N + N^2 + N^3 + ... + N^n$ trasformazioni per ottenere tutti i dettagli richiesti dalla rappresentazione grafica approssimata del frattale IFS: si tratta spesso di una mole di calcoli proibitiva per la potenza computazionale di un calcolatore convenzionale. Per superare questa difficoltà Barnsley ha proposto una tecnica alternativa - denominata "gioco del caso" - per la rappresentazione grafica dei frattali IFS. Nel gioco del caso viene scelto casualmente un punto del piano geometrico da utilizzarsi come "figura" iniziale per l'applicazione iterata delle trasformazioni del sistema, selezionate di volta in volta in modo casuale: se il numero di iterazioni è sufficientemente elevato l'immagine finale del punto appartiene al frattale attrattore. La ripetizione del procedimento per un adeguato numero di punti iniziali diversi permette di ottenere immagini realistiche come quella della foglia riportata nella **figura 8** il cui codice Fractint è il seguente:

```
Leaf {
 0.14    0.01      -0.00     0.51      -0.08     -1.31     0.06
 0.43    0.52      -0.45     0.50      1.49      -0.75     0.37
 0.45    -0.49     0.47      0.47      -1.62     -0.74     0.36
 0.49    -0.00     0.00      0.51      0.02      1.62      0.21
}
```

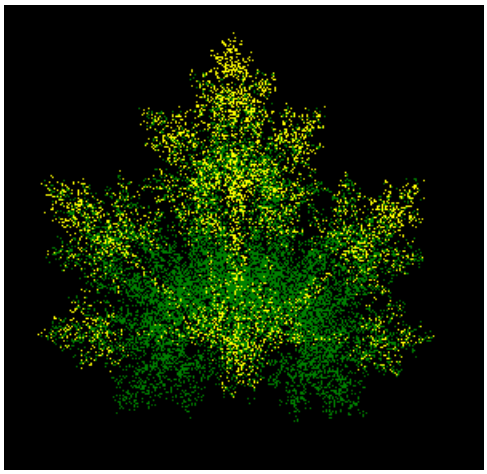


Figura 8

Per aumentare la velocità di convergenza del gioco del caso è importante selezionare le trasformazioni con una frequenza proporzionale all'area del frattale che contribuiscono a definire: l'ultima colonna della codifica IFS di Fractint esprime la probabilità con cui ogni trasformazione deve essere utilizzata nella costruzione dell'immagine. Si noti che i frattali IFS individuano il loro attrattore (l'immagine finale) in modo rigidamente deterministico e che il ruolo del caso in questa tecnica di rappresentazione è esclusivamente strumentale.

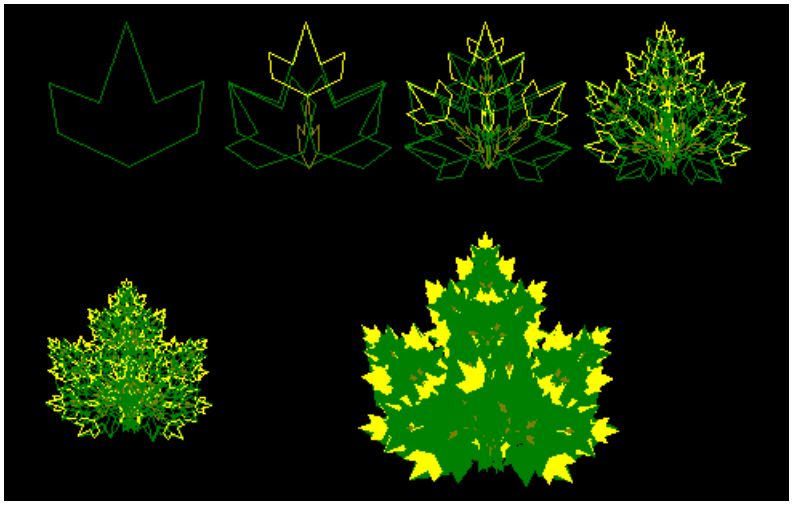


Figura 9

Nella **figura 9** sono rappresentate le prime 4 approssimazioni del precedente frattale IFS ottenute con la tradizionale tecnica deterministica delle trasformazioni iterate, in particolare l'ultima immagine è un *collage* ottenuto impiegando nel procedimento di costruzione figure interamente colorate.
La geometria "naturale" dei frattali biomorfi è indubbiamente affascinante, ma è davvero questo il segreto della stupefacente diversità di forme del mondo vivente?

Software e bibliografia

Fractint
Fractint è un diffusissimo programma *freeware* (è disponibile, oltre all'eseguibile, anche l'intero codice sorgente) di grafica frattale sviluppato da un gruppo di autori principali (*The Stone Soup Group*: Bert Tyler, Tim Wegner, Mark Peterson e Pieter Branderhorst) con il contributo telematico su *Compuserve* di decine di autori secondari (tra cui Adrian Mariano per gli *L-Systems* e Rob Beyer per gli *IFS*). La *contribution policy* di *The Stone Soup Group* (si può comprendere il significato di questo nome solo leggendo l'incredibile *Stone Soup Story* inclusa nella documentazione del programma) è molto semplice: *don't want money, got money, want admiration*!
La versione di Fractint di cui dispongo è un adattamento per Windows della versione per DOS e permette di rappresentare un vastissimo campionario di forme frattali in un *framework* sostanzialmente unitario: alla specificazione numerica degli estremi orizzontali e verticali, inferiori e superiori della finestra del piano geometrico di interesse per un particolare insieme frattale (*zoom*) segue una fase di calcolo dell'immagine ad un livello di dettaglio commisurato alla risoluzione disponibile per la visualizzazione. In questo modo un calcolo iterativo potenzialmente infinito viene interrotto una voltà raggiuntà la "profondità di dettaglio" necessaria.
Fractint, come ogni programma di rappresentazione grafica di insiemi frattali, è essenzialmente fondato su un ciclo di scansione dei singoli *pixels* disponibili per la visualizzazione: il colore del singolo pixel è calcolato con un ciclo di iterazione caratteristico del frattale stesso e, generalmente, rappresenta la "velocità di convergenza" dell'iterazione stessa (per una esposizione dettagliata relativa all'insieme frattale di Mandelbrot si può consultare l'articolo di Roberto Mariotti - "Indagine numerica sull'insieme di Mandelbrot" - pubblicato su Computer Programming n. 35, Aprile 1995; Fractint permette comunque di selezionare codifiche del colore alternative rispetto al criterio della "velocità di convergenza dell'iterazione").
Il nome Fract-int è giustificato dal prevalente ricorso del programma - con conseguente incremento della velocità di computazione - a calcoli in aritmetica di macchina con interi a 16 o 32 bit in sostituzione dell'aritmetica *floating-point* propria delle formule che caratterizzano gli insiemi frattali. Questo aspetto del programma è descritto - come tutti gli altri dettagli tecnici, spesso ingegnosi ed istruttivi - nel libro di Tim Wegner e Mark Peterson: *Fractal Creation*, Waite Group Press, Mill Valley - California, 1991.
Le figure 1, 3, 4, 5 e 8 sono state realizzate con Fractint.

Fractal Vision
Fractal Vision della Cedar Software è un programma *shareware* per la realizzazione di immagini frattali con codifica IFS. Fractal Vision è stato realizzato da Dick Oliver della Cedar Software e portato in ambiente Windows da Daniel Hoviss e Al Bupp della Dosolutions.
Il programma può leggere e scrivere file .ifs compatibili con il formato Fractint, ma in Fractal Vision le equazioni che caratterizzano l'insieme delle trasformazioni di un frattale IFS sono definite e rappresentate graficamente! Per definire un nuovo IFS è infatti sufficiente disegnare una figura iniziale (*seed*) e un numero di figure trasformate più piccole (*parts*) ognuna delle quali individua una delle trasformazioni del sistema. Una figura *seed* e il corrispondente insieme di *parts* costituiscono un modello completo di un frattale IFS (*template*). Nella prima immagine della figura 7 il *seed* è in rosso, mentre le *parts* sono in giallo: insieme costituiscono un *template* che definisce il triangolo di Sierpinski. Una volta impostato il livello di profondità desiderato per l'iterazione è possibile rappresentare graficamente un frattale a partire dal relativo *template* con il comando *draw* (tecnica di rappresentazione deterministica) o con il comando *paint* (tecnica di rappresentazione casuale di Barnsley).
Le figure 2, 6, 7 e 9 sono state realizzate con Fractal Vision.

[1] Mandelbrot B. B., "The Fractal Geometry of Nature", Freeman, 1982
[2] Dewdney A. K., "Monti frattali, piante graftali e grafica al calcolatore", Le Scienze (ed. it. di *Scientific American*) n. 222, Febbraio 1987
[3] Barnsley M., "Fractals Everywhere", Academic Press, 1988

La più accessibile introduzione alla geometria frattale che io conosca è la seguente:

> Marco d'Eramo, "Nei meandri dei frattali", in: Aa.Vv., "Gli ordini del caos", ed. manifestolibri, 1991

la seguente raccolta di articoli:

> Giulio Casati (a cura di), "Il caos: le leggi del disordine", ed. Le Scienze, 1991

contiene una ristampa del citato articolo di Dewdney **[2]**, dove si introducono gli L-system, ed anche la ristampa di un articolo molto interessante - scritto da Hartmut Jurgens, Heinz-Otto Peitgen e Dietmar Saupe - dal titolo "Il linguaggio dei frattali", originalmente pubblicato sulla rivista "Le Scienze" (ed. it. di *Scientific American*), n. 266 dell'Ottobre 1990.

Il lettore non si lasci trarre in inganno dal fatto che si tratta di pubblicazioni relative a studi nel campo dei sistemi caotici: i frattali sono la geometria del caos! A questo proposito si veda anche:

> Giorgio Meini, "Studio del comportamento caotico e frattale di una semplice funzione", Computer Programming, n. 32, Gennaio 1995

Il seguente è infine un articolo molto leggibile che descrive l'uso della codifica IFS per costruire modelli frattali di forme naturali:

> Paolo Sommaruga, "Modelli frattali di oggetti naturali", Le Scienze (ed. it. di *Scientific American*), n. 282, Febbraio 1992

Listato 1

```
Leaf1 { ; Adrian Mariano, from the Algorithmic Beauty of Plants
        ; Compound leaf with alternating branches, Figure 5.12b p.130
  angle 8
  axiom x
  a=n
  n=o
  o=p
  p=x
  b=e
  e=h
  h=j
  j=y
  x=F[+A(4)]Fy
  y=F[-B(4)]Fx
  F=@1.18F@i1.18
  }

Leaf2 { ; Adrian Mariano, from the Algorithmic Beauty of Plants
        ; Compound leaf with alternating branches, Figure 5.12a p.130
  angle 8
  axiom a
  a=f[+x]fb
  b=f[-y]fa
  x=a
  y=b
  f=@1.36f@i1.36
  }

Plant01 { ; Adrian Mariano, from the Algorithmic Beauty of Plants
          ; Plant-like structure, figure 1.24a p.25
  angle 14
  axiom f
  f=F[+F]F[-F]F
  }

Plant02 { ; Adrian Mariano, from the Algorithmic Beauty of Plants
          ; Plant-like structure, figure 1.24b p.25
  angle 18
  axiom f
  f=F[+F]F[-F][F]
  }

Plant03 { ; Adrian Mariano, from the Algorithmic Beauty of Plants
          ; Plant-like structure, figure 1.24c p.25
  angle 16
  axiom f
  f=FF-[-F+F+F]+[+F-F-F]
  }
```

```
Plant04 { ; Adrian Mariano, from the Algorithmic Beauty of Plants
          ; Plant-like structure, figure 1.24d p.25
  angle 18
  axiom x
  X=F[+X]F[-X]+X
  F=FF
  }

Plant05 { ; Adrian Mariano, from the Algorithmic Beauty of Plants
          ; Plant-like structure, figure 1.24e p.25
  angle 14
  axiom x
  X=f[+X][-X]FX
  F=FF
  }

Plant06 { ; Adrian Mariano, from the Algorithmic Beauty of Plants
          ; Plant-like structure, figure 1.24f p.25
  angle 16
  axiom x
  X=F-[[X]+X]+F[+FX]-X
  F=FF
  }

Tree { ; Adrian Mariano
       ; from The Fractal Geometry of Nature by Mandelbrot
  angle=12;
  axiom +++FX
  X=@.6[-FX]+FX
  }
```

Listato 2

```
Fern {
0.0         0.0         0.0         .17         0.0         0.0         .05
.84962      .0255       -.0255      .84962      0.0         3           .75
 -.1554     .235        .19583      .18648      0.0         1.2         .1
 .1554      -.235       .19583      .18648      0.0         3           .1
}

Leaf {
 0.14 0.01  -0.00 0.51  -0.08 -1.31 0.06
 0.43 0.52  -0.45 0.50  1.49  -0.75 0.37
 0.45 -0.49 0.47  0.47  -1.62 -0.74 0.36
 0.49 -0.00 0.00  0.51  0.02  1.62  0.21
}

Tree {
.195  -.488 .344  .443  .722  .536  .4
.462  .414  -.252 .361  .538  1.167 .4
-.058 -.070 .453  -.111 1.125 .185  .1
-.045 .091  -.469 -.022 .863  .871  .1
}
```

Volpi e conigli: studio di un sistema predatore-preda

Giorgio Meini

I classici sistemi di equazioni differenziali impiegati in biologia nello studio della relazione ecologica tra predatore e preda non sono un modello affidabile per un semplice "mondo" simulato al calcolatore

Il programma del **listato 1** (riportato in fondo all'articolo) è stato sviluppato nei laboratori dell'I.T.I. di Siena quasi dieci anni fa come esercizio di programmazione in linguaggio Pascal (in ambiente UNIX) dai miei studenti di allora. Ho effettuato la conversione in Turbo-Pascal (versione 6.0 per DOS) attenendomi scrupolosamente ai più rigidi criteri filologici: il programma rimane essenziale sotto l'aspetto grafico e non prevede modalità di interazione con l'utente (ogni variazione dei parametri - dichiarati come CONST - richiede infatti una nuova compilazione). Nell'attesa che qualcuno dei lettori realizzi una versione aggiornata a 32 bit per Windows 95 questo vecchio programma è comunque ancora in grado di simulare per noi la spietata lotta ecologica tra squali (predatori) ed altri pesci (prede) che si svolge quotidianamente nelle acque dei mari che ricoprono l'intera superficie del pianeta toroidale Wa-Tor (**figura 1**).

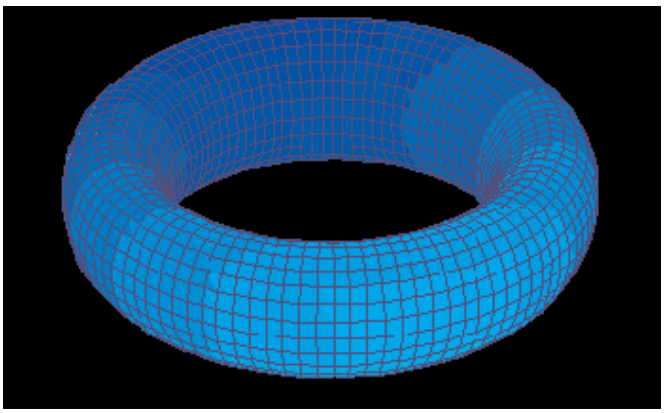
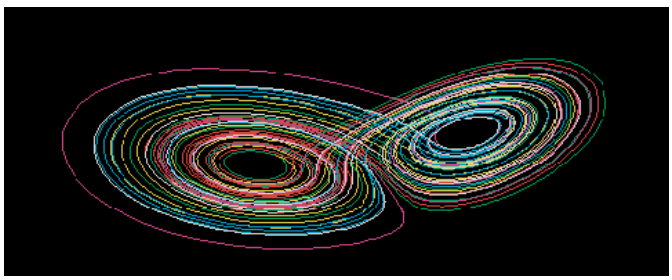


Figura 1 - Il pianeta toroidale Wa-Tor è completamente ricoperto dalle acque marine.

Il programma è una fedele implementazione delle regole esposte da Dewdney sulla rivista "Le Scienze" del Febbraio 1985 [1]:

- lo spazio del gioco è una griglia rettangolare di caselle avente i lati opposti adiacenti (toro);



- il tempo consiste in una successione di istanti discreti denominati crononi;
- in ogni cronone di tempo ciascun pesce si sposta in modo casuale dalla casella che occupa ad una delle 8 caselle adiacenti, se nell'intorno non esistono caselle vuote il pesce non si sposta;
- in ciascun cronone di tempo ogni squalo mangia uno dei pesci che si trovano nelle caselle adiacenti scegliendo casualmente la preda, se non esistono pesci nelle caselle dell'intorno lo squalo si sposta in modo casuale in una di queste 8 caselle;
- pesci e squali si riproducono periodicamente (il periodo riproduttivo dei pesci è diverso da quello degli squali), ma l'età riproduttiva varia da pesce a pesce e da squalo a squalo;
- se uno squalo non mangia muore dopo un numero prefissato di crononi;
- il gioco inizia con un numero prefissato di squali e di pesci distribuiti casualmente in tutto lo spazio del gioco;
- per chi preferisce le volpi e i conigli agli squali e ai pesci è sufficiente indirizzare la propria immaginazione a prati e zampe anziché a mari e pinne.

Nonostante la semplice rappresentazione alfanumerica (**figura 2**) seguire l'evoluzione del gioco nel tempo è affascinante: le due popolazioni - squali e pesci - variano continuamente la loro consistenza numerica e la loro distribuzione spaziale. È anche possibile che una delle due popolazioni si estingua: nel caso di estinzione degli squali i

pesci - rimasti privi di predatori - occuperanno in breve tempo tutto lo spazio del gioco, l'estinzione dei pesci causa invece anche quella degli squali per la cui sopravvivenza è determinante la presenza di prede.

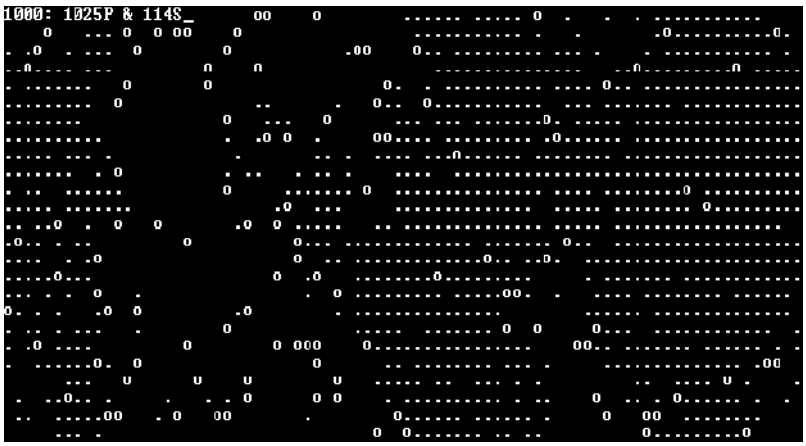


Figura 2 - La disposizione dei pesci (.) e degli squali (o) sulla superficie del pianeta Wa-Tor al termine di una simulazione di 1000 cronomi.

L'esistenza di scelte casuali nelle regole fa sì che la simulazione della vita sul pianeta Wa-Tor sia a priori un gioco non deterministico. Ma dopo avere osservato alcune simulazioni è facile convincersi che questa componente aleatoria del gioco ha conseguenze trascurabili sull'equilibrio a lungo termine dell'ecosistema Wa-Tor: questo infatti dipende quasi esclusivamente dalle reciproche relazioni tra alcuni parametri quali le dimensioni dello spazio del gioco, i periodi riproduttivi dei pesci e degli squali, la resistenza alla fame da parte degli squali e le dimensioni delle due popolazioni iniziali.

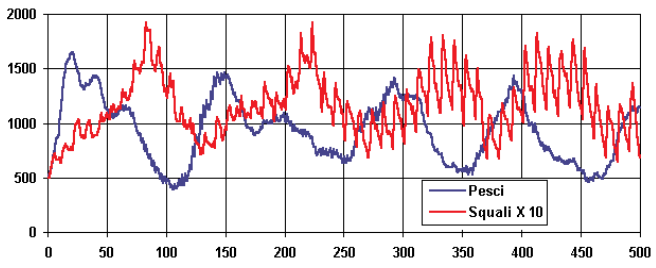


Figura 3 - Grafico della variazione del tempo delle popolazioni dei pesci e degli squali in una simulazione di 500 crononi (il valore dei parametri è riportato nel testo)

La **figura 3** rappresenta la variazione nel tempo delle due popolazioni per una scelta dei parametri (dimensioni dello spazio di gioco: 80 x 25 caselle, numero iniziale di pesci: 500, numero iniziale di squali: 50, periodo di riproduzione pesci: 3 crononi, periodo di riproduzione squali: 10 crononi, resistenza alla fame da parte degli squali: 3 crononi) che garantisce, almeno apparentemente, una lunga convivenza di entrambe le specie (le rapide e pronunciate oscillazioni del grafico relativo alla popolazione degli squali risultano amplificate dall'applicazione di un coefficiente moltiplicativo ai dati numerici). Sembra che si stabilisca una sorta di equilibrio ciclico tra le due specie:

- l'aumento del numero dei pesci è causa di un successivo incremento della popolazione degli squali che se ne cibano;
- con la crescita del numero degli squali aumenta il fabbisogno di prede e, di conseguenza, diminuisce il numero dei pesci;
- una ridotta popolazione di pesci causa la morte per inedia di alcuni predatori: il numero degli squali diminuisce;
- la minore pressione predatoria esercitata da una diminuita popolazione di squali permette infine ai pesci di riprodursi e di incrementare il proprio numero.

Questa forma dinamica di stabilità strutturale è indubbiamente una conseguenza delle condizioni iniziali, tanto che una modifica nel valore di uno o più parametri è spesso motivo di una rapida estinzione di una o di entrambe le popolazioni. È possibile prevedere una situazione di equilibrio ciclico basandosi esclusivamente sul valore dei parametri? Un programma di simulazione ci permette di osservare - ma non di prevedere - l'evoluzione di un ipotetico sistema e quando piccole differenze nel valore dei parametri causano comportamenti qualitativamente diversi non è facile estrapolare una tendenza nell'evoluzione del sistema stesso. Dare una risposta alla domanda è quindi cosa tutt'altro che semplice: un approccio classico al problema consiste nel modellare il sistema predatore-preda con un sistema di due equazioni differenziali del primo ordine (si veda il riquadro a parte per una breve introduzione alle equazioni differenziali, proposta in coda all'articolo).

La crescita di una popolazione priva di predatori che disponga di infinite risorse alimentari ed ambientali è descritta dall'equazione differenziale $x'(t) = rx$ avente come soluzione la funzione esponenziale $x = x_0 e^{rt}$ (r è il fattore di accrescimento e x_0 la popolazione iniziale) il cui grafico è riportato in nero nella **figura 4** (realizzata con *MapleV*). Ma, nonostante l'economia umana ancora non lo riconosca, il concetto di "risorse limitate" è un principio fondamentale dell'ecologia moderna: l'equazione differenziale logistica $x'(t) = rx(1-x/k)$ - introdotta da Lotka nel 1925 - pur avendo un comportamento iniziale esponenziale rappresenta la limitatezza delle risorse disponibili per l'accrescimento con il parametro k (capacità portante). La soluzione dell'equazione logistica (calcolata con *Mathematica*) è

$$x(t) = k/(1-(1-k/x_0)e^{-rt})$$

ed è rappresentata in rosso nel grafico della figura 4: è evidente come la popolazione x evolva con il tempo verso uno stato di equilibrio corrispondente alla capacità portante k. [2]

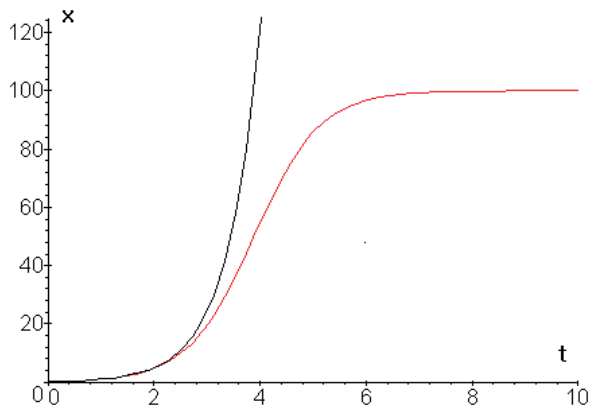


Figura 4 - Grafico delle leggi di crescita esponenziale (in nero) e logistica (in rosso).

Un sistema di equazioni differenziali del primo ordine permette di descrivere le relazioni che sussistono tra due grandezze che variano nel tempo - in questo caso le dimensioni delle popolazioni dei predatori e delle prede - e tra le loro velocità di variazione (le rispettive derivate prime). Un modello matematico della relazione preda-predatore (noto come equazioni di Volterra, dal nome del matematico Vito Volterra vissuto a cavallo tra il XIX e il XX secolo) è dato dal seguente sistema:

$$\begin{cases} x'(t) = ax - bxy \\ y'(t) = -cy + dxy \end{cases}$$

dove a, b, c e d sono parametri reali positivi che caratterizzano quantitativamente il comportamento del modello. In assenza di predatori (y(t) = 0) la popolazione delle prede (x) segue la legge di accrescimento esponenziale x′(t) = ax, mentre in assenza di prede (x(t) = 0) i predatori (y) si estinguono con l'andamento esponenziale negativo dato da y′(t) = -cy. L'ipotesi fondamentale del modello è che l'interazione tra le due specie ne influenzi la variazione in modo proporzionale al prodotto tra il numero di prede e il numero di predatori presenti ad un dato istante secondo i coefficienti -b (diminuzione delle prede) e d (aumento dei predatori). Sia *Mathematica* che *MapleV* sono in grado di risolvere molti sistemi di equazioni differenziali, ma entrambi falliscono nel tentativo di fornire una soluzione per il sistema di Volterra: in questo caso però il motivo non è da ricercarsi in una limitazione di questi programmi, ma piuttosto nel fatto che le funzioni x(t) e y(t) che risolvono il sistema non sono rappresentabili in termini di funzioni elementari e, di conseguenza, non sono determinabili per via analitica. L'inutilità delle tecniche del calcolo simbolico nella ricerca di una soluzione esatta rende in questo caso necessario il ricorso a procedure di calcolo numerico per ottenere una soluzione approssimata (per una breve introduzione all'argomento si legga il riquadro relativo proposto al termine dell'articolo).
In una prima fase dello studio orientata ad una interpretazione qualitativa - e non quantitativa - dei risultati è possibile evitare di determinare il valore dei parametri a, b , c, d con l'accuratezza che sarebbe necessaria per modellare con precisione il comportamento del sistema in esame. Ponendo i parametri a = 0.25, b = 0.5, c = 0.125, d = 0.5 e le condizioni iniziali x(0) = 0.5 e y(0) = 0.1 (le dimensioni delle popolazioni si intendono normalizzate rispetto a dimensioni di riferimento) i seguenti comandi *MapleV* permettono di realizzare il grafico della **figura 5** dove è rappresentato l'andamento temporale delle popolazioni delle prede e dei predatori:

```
sol:=dsolve({diff(x(t),t)=
.25*x(t)-.5*x(t)*y(t),
diff(y(t),t)=.125*y(t)+.5*x(t)*y(t),
                    x(0)=.5,y(0)=.1},{x(t),y(t)},
              type=numeric,output=listprocedure);
prey:=subs(sol,x(t));
predator:=subs(sol,y(t));
plot({prey,predator},0..125);
```

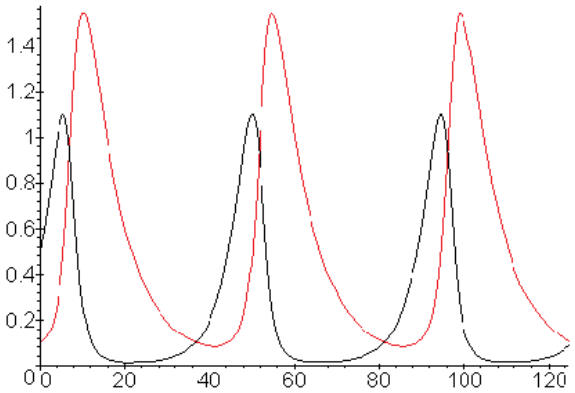


Figura 5 - Soluzione del sistema di equazioni differenziali di Volterra: rappresentano la variazione nel tempo delle popolazioni delle prede e dei predatori.

La periodicità della soluzione è congruente con l'andamento ciclico dei dati ottenuti dalla simulazione; allo scopo di evidenziare la relazione predatore-preda è possibile tracciare un grafico parametrico di x(t) e di y(t) con parametro comune t:

```
plot([prey,predator,0..125],0..1.5,0..1.75);
```

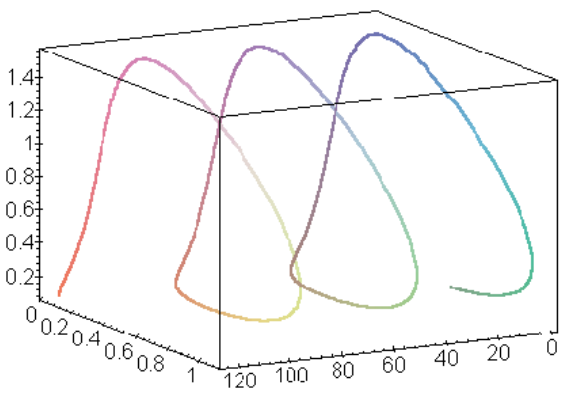


Figura 6 - Traiettoria nello spazio delle fasi della soluzione del sistema di Volterra.

Nella **figura 6** è riportato il grafico che si ottiene e che rappresenta un esempio di traiettoria nello spazio delle fasi di un sistema dinamico: in questo caso l'attrattore (la situazione verso la quale evolve il sistema) è costituito da un ciclo che lo stato del sistema (determinato dalle dimensioni variabili delle due popolazioni) percorre nel corso della sua evoluzione temporale. Se si pensa alla soluzione del sistema di Volterra come una curva nello spazio tridimensionale definito dalle dimensioni delle due popolazioni e dal tempo (**figura 7** realizzata con il comando **odeplot** del *package* **plots** di *MapleV*) allora sia i grafici relativi alla variazione temporale delle popolazioni delle prede e dei predatori, che il diagramma rappresentante lo spazio delle fasi del sistema non sono altro che particolari proiezioni bidimensionali della stessa curva.

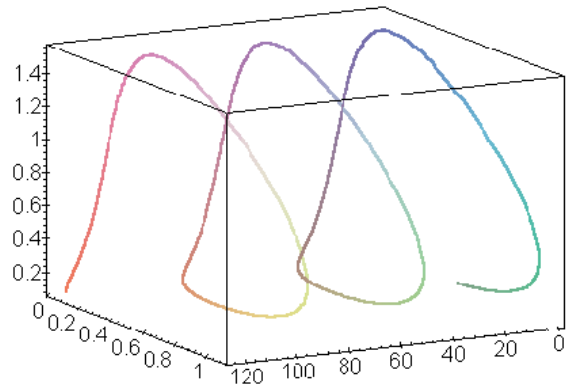


Figura 7 - Traiettoria tridimensionale (preda, predatore, tempo) della soluzione del sistema di Volterra.

Non si deve però dimenticare che il comportamento del sistema predatore-preda è determinato dalle dimensioni iniziali delle due popolazioni e, al fine di studiare l'influenza della situazione iniziale, è possibile rappresentare nel medesimo diagramma di fase traiettorie corrispondenti a condizioni iniziali diverse.

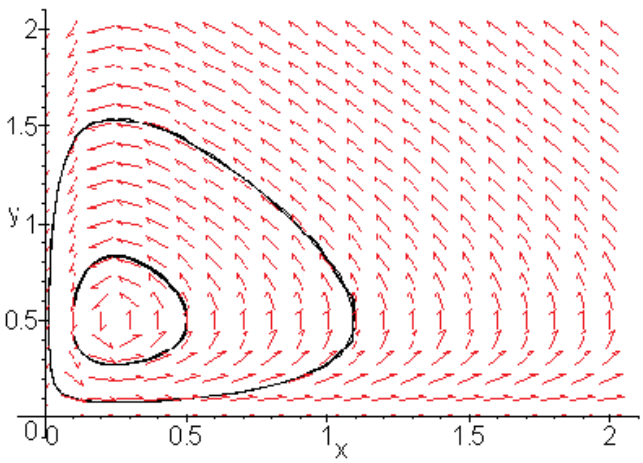


Figura 8 - Traiettorie corrispondenti a condizioni iniziali diverse e campo direzionale del sistema di equazioni differenziali di Volterra.

La **figura 8** è stata realizzata con il comando **DEplot2** del *package* **DEtools** di *MapleV* e rappresenta le traiettorie corrispondenti a tre diverse condizioni iniziali (x(0)=0.5, y(0)=0.1; x(0)=0.5, y(0)=0.5; x(0)=0.1, y(0)=0.5), due delle quali convergono praticamente allo stesso attrattore; in ogni caso l'attrattore è sempre costituito da un ciclo, situazione che riflette la periodicità delle soluzioni (cioè della variazione nel tempo delle dimensioni delle due popolazioni). Sullo sfondo del grafico della figura 8 è riportato in rosso il campo direzionale del sistema di equazioni differenziali di Volterra: le frecce indicano la direzione della tangente ad una ipotetica soluzione (dipendente da particolari condizioni iniziali) che si trovasse a passare per quel punto e rendono evidente come in ogni caso - anche variando le condizioni iniziali - la traiettoria del sistema sia un ciclo a cui corrispondono soluzioni periodiche. Le uniche eccezioni si hanno per condizioni iniziali dove x(0)=0 - in questo caso l'assenza permanente di prede causa una estinzione dei predatori: nel grafico infatti le frecce giacenti sull'asse y, corrispondenti a valori nulli di x, sono dirette verso il punto dove sia x che y valgono zero - o dove y(0)=0, per cui l'assenza di predatori è causa di un incremento illimitato delle prede: infatti nel grafico le frecce giacenti sull'asse x - valori nulli di y - sono dirette verso valori infiniti di x e nulli di y. Si può quindi affermare che il modello di Volterra per il sistema predatore-preda non è in grado di prevedere l'estinzione di una delle due specie, ma è proprio questa invece la situazione che si verifica per molte e diverse scelte dei valori iniziali delle due popolazioni nella simulazione condotta con il programma Wa-Tor.

Esiste un modello differenziale alternativo della relazione preda-predatore - noto come sistema di Lotka-Volterra - per cui, in assenza di predatori, la popolazione delle prede cresce secondo la legge logistica senza superare una dimensione massima imposta dalla limitata disponibilità delle risorse ambientali ed alimentari (la popolazione dei predatori invece continua a dipendere da quella delle prede come nel sistema di Volterra):

$$\begin{cases} x'(t) = ax - kx^2 - bxy \\ y'(t) = -cy + dxy \end{cases}$$

Questa caratteristica del sistema di Lotka-Volterra riflette il vincolo costituito dal limitato spazio a disposizione per l'accrescimento sul pianeta Wa-Tor. Le **figure 9, 10, 11** e **12** sono da confrontarsi, rispettivamente, con le figure 5, 6, 7 e 8: esse rappresentano per il sistema di Lotka-Volterra i risultati dello stesso studio precedentemente condotto, con l'ausilio di *MapleV*, per il sistema di Volterra (per consentire un confronto tra i due modelli, i valori dei parametri a, b, c e d sono gli stessi e il valore del parametro k è fissato in 0.125 in modo che la capacità portante a/k, corrispondente alla dimensione massima normalizzata consentita per la popolazione delle prede, risulti essere uguale a 2.0).

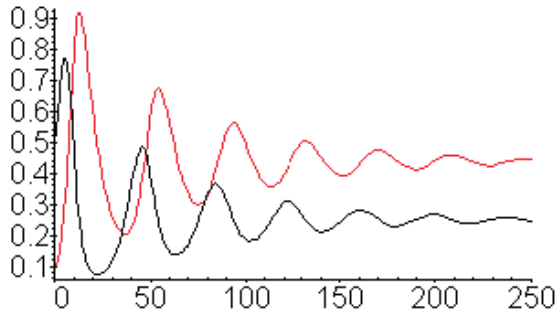


Figura 9 - Soluzione del sistema di equazioni differenziali di Lotka-Volterra: rappresentano la variazione nel tempo delle popolazioni delle prede e dei predatori.

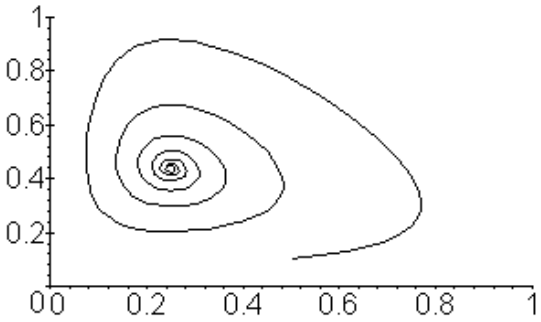


Figura 10 - Traiettoria nello spazio delle fasi della soluzione del sistema Lotka-Volterra.

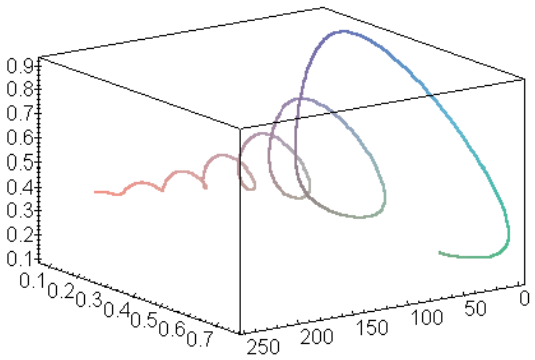


Figura 11 - Traiettoria tridimensionale (preda, predatore, tempo) della soluzione del sistema Lotka-Volterra.

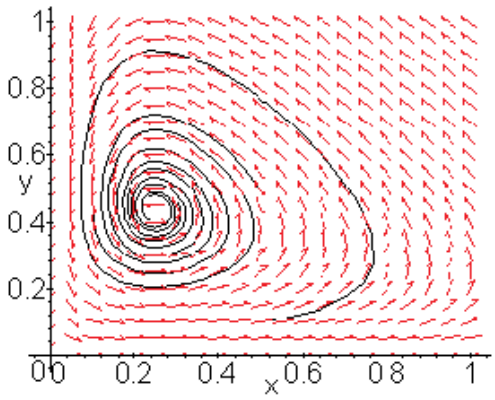


Figura 12 - Traiettorie corrispondenti a condizioni iniziali diverse e campo direzionale del sistema di equazioni differenziali di Lotka-Volterra.

In questo caso le oscillazioni nella variazione delle dimensioni delle due popolazioni si smorzano progressivamente con il trascorrere del tempo evolvendo verso una situazione limite (asintotica) in cui le popolazioni convivono mantenendosi costanti (figura 9): l'attrattore consiste quindi in un punto nello spazio delle fasi del sistema (figura 10) le cui coordinate rappresentano i valori di equilibrio stabile ed asintotico per x (prede) ed y (predatori). Come risulta dal grafico della figura 12 questo comportamento caratteristico non dipende dalle condizioni iniziali (le dimensioni iniziali delle popolazioni delle due specie) e, se confrontato con i risultati della simulazione riportati nel grafico della figura 3, non costituisce un valido modello ecologico per il pianeta Wa-Tor. È interessante confrontare gli attrattori dei due modelli (figure 6 e 10) con la traiettoria nello spazio delle fasi tracciata in base ai risultati della simulazione e riportata nella **figura 13** (realizzata con *Mathematica* della Wolfram Research e rappresentante la traiettoria nello spazio delle fasi relativa ai risultati della simulazione riportati nel grafico della figura 3; il programma di simulazione è stato modificato nella parte di scrittura dei dati su file al fine di renderne la lettura compatibile con il comando READLIST di Mathematica anziché con la funzione IMPORTA DATI di Microsoft Draw).

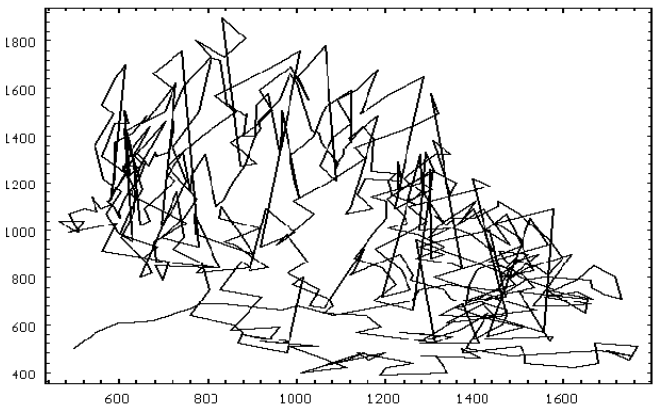


Figura 13 - Grafico realizzato con Mathematica della Wolfram Research e rappresentante le traiettorie nello spazio delle fasi relative ai risultati della simulazione riportati nel grafico della figura 3. Il programma di simulazione è stato modificato nella parte di scrittura dei dati su un file al fine di rendere la lettura compatibile con il comando READLIST di Mathematica anziché la funzione IMPORTA DATI di Microsoft Draw.

L'evoluzione del sistema si svolge in una zona delimitata dello spazio delle fasi, ma senza convergere ad una situazione di equilibrio stazionario o dinamico: l'attrattore è in questo caso un "attrattore strano" (per una introduzione "ricreativa" a questo concetto si veda [3]). Uno dei primi attrattori strani oggetto di studio è stato l'attrattore frattale di Lorenz la cui immagine è riportata in apertura di questo articolo; esso rappresenta la soluzione nello spazio delle fasi tridimensionale di un sistema di tre equazioni differen-

ziali del primo ordine che descrivono il comportamento di un volume circoscritto di atmosfera [di fatto Poincaré ha dimostrato, alla fine del secolo scorso, che un sistema di due equazioni differenziali del primo ordine non può avere una soluzione convergente ad un "attrattore strano"; il semplice caso preso in esame in [2] non si riferisce ad una equazione differenziale continua, ma ad una equazione alle differenze discreta, N.d.A.]:

$$\begin{cases} x' = a(y - x) \\ y' = bx - y - xz \\ z' = -cz + xy \end{cases}$$

L'importanza degli attrattori strani nello studio della realtà naturale ed artificiale è confermata dal successo interpretativo delle nuove discipline che hanno come oggetto di studio l'evoluzione caotica dei sistemi dinamici non lineari. Per esempio, la rappresentazione in un particolare diagramma di fase (autospazio) della frequenza cardiaca di un cuore sano costituisce un attrattore strano, mentre un attrattore di tipo ciclico (cui corrisponde una elevata regolarità della frequenza cardiaca) è in questo caso indice di una possibile cardiopatia [4].

Possiamo concludere affermando il fallimento dei sistemi di equazioni differenziali - ai quali classicamente si ricorre nello studio della relazione predatore-preda - nel descrivere il semplice ecosistema marino del pianeta Wa-Tor? Non ancora:! Lo studio condotto è infatti fondato sulla modificazione delle sole condizioni iniziali, mentre la variazione dei parametri disponibili per la simulazione si riflette anche sul valore delle costanti (a, b, c, d e k) dei modelli presi in esame. Il lettore intenzionato ad approfondire questo aspetto con l'ausilio di *MapleV* troverà nel **listato 2** (proposto in fondo all'articolo) tutti i comandi utilizzati per la produzione dei grafici che compaiono in questo articolo: è sufficiente modificare il valore dei parametri costanti.

Un programma come *MapleV* (o come *Mathematica* per il quale non è difficile adattare i comandi del listato 2) permette di utilizzare potenti strumenti matematici concentrando l'attenzione sui risultati che producono, ma ignorando le complesse tecniche risolutive necessarie alla loro manipolazione; per esempio: il grafico della figura 14 rappresenta l'attrattore frattale di Lorenz (a=10, b=28, c=8/3, x(0)=y(0)=z(0)=5) ed è stato ottenuto con un solo comando del *package* **Detools** di *MapleV*:

```
DEplot([10*(y-x),28*x-y-x*z,-(8/3)*z+x*y],
    [x,y,z],t=0..50,{[0,5,5,5]},stepsize=.05);
```

L'attrattiva della simulazione della vita su Wa-Tor consiste, oltre che nella quasi-ciclica variazione del numero di pesci e di squali presenti, anche nella continua modificazione della loro disposizione spaziale: qualcuno tra i lettori è in grado di trovare un metodo (matematico o informatico) che permetta di avanzare congetture o previsioni riguardo a questo aspetto del gioco? E non dimenticate che la domanda iniziale riguardo alla possibilità di prevedere una situazione di equilibrio ciclico (delle dimensioni delle popolazioni di prede e predatori) basandosi sul valore dei parametri è ancora priva di una risposta soddisfacente.

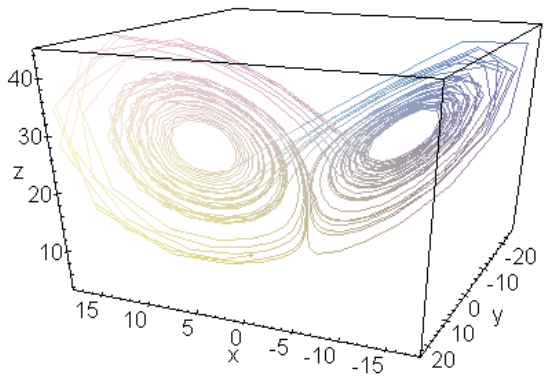


Figura 14 - Grafico realizzato con Maple V: rappresenta l'attrattore caotico e frattale tridimensionale del sistema di equazioni differenziali di Lorenz.

Software, riferimenti e indicazioni bibliografiche

[1] A. Dewdney, "Squali ed altri pesci combattono una guerra ecologica sul pianeta Wa-Tor", *Le Scienze* (ed. it. di *Scientific American*), Febbraio 1985
[2] G. Meini, "Studio del comportamento caotico e frattale di una semplice funzione", *Computer Programming* n. 32, Gennaio 1995
L'articolo precedente comprende anche una breve presentazione del programma *Mathematica* della Wolfram Research.
[3] D. Hofstadter, "Attrattori strani: enti fra ordine e caos", *Le Scienze* (ed. it. di *Scientific American*), Febbraio 1982
[4] A. Goldberger, D. Rigney, B. West, "Caos e frattali in fisiologia umana", *Le Scienze* (ed. it. di *Scientific American*), Aprile 1990
Gli articoli [3] e [4] sono stati ristampati in:
G. Casati (a cura di), "Il caos: le leggi del disordine", *Le Scienze* ed., 1991

Per una introduzione tecnica alla teoria generale dei sistemi dinamici, o per un approfondimento dei concetti di equilibrio e stabilità nell'evoluzione di un sistema dinamico, si veda:
S. Rinaldi, *Teoria dei sistemi*, ed. Clup, Milano

Il seguente è un testo che tratta essenzialmente dei modelli matematici (prevalentemente differenziali) dei sistemi predatore-preda, partendo dal semplice sistema di Volterra e introducendo successivamente concetti quali: la struttura di età di una popolazione, l'effetto dei nascondigli per le prede e la distribuzione sul territorio dei predatori. L'autore ha limitato le difficoltà matematiche preferendo, quando possibile, una interpretazione qualitativa dei modelli e rimandando alle appendici per un approfondimento di tipo quantitativo. La data di pubblicazione lo rende un testo classico, ma del tutto mancante di riferimenti ai modelli di simulazione per calcolatore (oggi molto diffusi) sia deterministici che probabilistici e alle conseguenze dell'esistenza di attrattori caotici la cui importanza è stata inizialmente evidenziata proprio da studiosi di biologia teorica come Robert May.
M. Smith, *L'ecologia e i suoi modelli*, ed. Scientifiche e Tecniche Mondadori, 1975
Nel VI capitolo di questo bellissimo libro (complesso, ma leggibile da tutti), gli autori interpretano la condizione di equilibrio dinamico che si stabilisce nell'evoluzione di un sistema predatore-preda come un caso particolare di processi di genesi e conservazione della forma energeticamente dissipativi. Nei capitoli XI e XII prendono invece in esame le conseguenza di varie modalità di crescita per le popolazioni.
M. Eigen & R. Winkler, *Il Gioco: le leggi naturali governano il caso*, Adelphi ed., 1986

LISTATO 1

```
  PROGRAM Wa_Tor;

{
 Simulazione di un sistema predatore-preda:
squali (o) e
 pesci (.) nel mare di un pianeta toroidale
(Wa-Tor).
 Riferimento: Dewdney in "Le Scienze" del
Febbraio 1985.
}

 USES Crt;

 LABEL Fine;

 CONST
  {
   Dimensioni schermo
  }
  Xmax=79;
  Ymax=24;
  {
   Parametri simulazione
  }
  TempoEsperimento=500; { "crononi" di tempo }
  NumPesciIniziali=500;
  NumSqualiIniziali=50;
  ProlificazionePesci=3; { crononi }
  ProlificazioneSquali=10; { crononi }
  InediaSquali=3; { crononi }

 TYPE
  TipoX = 0..Xmax; TipoY = 0..Ymax;

  TipoCella = RECORD
     case Cella: (Nulla, Pesce, Squalo) of
      Nulla: ();
      Pesce: (
               EtaPesce: 0..TempoEsperimento;
               PesceMosso: BOOLEAN
              );
      Squalo: (
                EtaSqualo: 0..TempoEsperimento;
                 SqualoMosso: BOOLEAN;
                FameSqualo: 0..(InediaSquali+1)
               )
    END;

  IntornoCella = (W, NW, N, NE, E, SE, S, SW);

 VAR
  DatiPesci: ARRAY [0..TempoEsperimento]
              OF 0..((Xmax+1)*(Ymax+1));
  DatiSquali: ARRAY [0..TempoEsperimento]
               OF 0..((Xmax+1)*(Ymax+1));
  Mondo: ARRAY [TipoX, TipoY] OF TipoCella;
  x,newX: TipoX; y,newY: TipoY; { coordinate }
  CelleLibere, CellePesce: ARRAY [IntornoCella]
                            OF BOOLEAN;
  NumCelleLibere, NumCellePesce: 0..8;
  Scelta: 1..8; Conta: 0..8;
  Pos: IntornoCella;
  ContaPesci, ContaSquali:
0..((Xmax+1)*(Ymax+1));
  Tempo: 0..TempoEsperimento;
   FileDati: TEXT;
   i: WORD;

 FUNCTION DecX (x: TipoX): TipoX;
  BEGIN
   DecX:=((Xmax+1)+x-1) MOD (Xmax+1)
  END;

 FUNCTION IncX (x: TipoX): TipoX;
  BEGIN
   IncX:=((Xmax+1)+x+1) MOD (Xmax+1)
  END;

 FUNCTION DecY (y: TipoY): TipoY;
  BEGIN
   DecY:=((Ymax+1)+y-1) MOD (Ymax+1)
  END;

 FUNCTION IncY (y: TipoY): TipoY;
  BEGIN
   IncY:=((Ymax+1)+y+1) MOD (Ymax+1)
  END;

 BEGIN
  ASSIGN(FileDati,'c:\temp\dati.txt');
  RANDOMIZE;
  CLRSCR;
  {
   Inizializzazione
  }
  FOR x:=0 TO Xmax DO
   FOR y:=0 TO Ymax DO
    Mondo[x,y].Cella:=Nulla;
  FOR i:=1 TO NumPesciIniziali DO
   BEGIN
    REPEAT
     x:=RANDOM((Xmax+1)); y:=RANDOM((Ymax+1))
    UNTIL Mondo[x,y].Cella = Nulla;
    Mondo[x,y].Cella:=Pesce;

Mondo[x,y].EtaPesce:=RANDOM(ProlificazionePesci+1);
    Mondo[x,y].PesceMosso:=FALSE;
    GOTOXY(x+1,y+1); WRITE('.');
   END;
  FOR i:=1 TO NumSqualiIniziali DO
   BEGIN
    REPEAT
     x:=RANDOM((Xmax+1)); y:=RANDOM((Ymax+1))
    UNTIL Mondo[x,y].Cella = Nulla;
    Mondo[x,y].Cella:=Squalo;

Mondo[x,y].EtaSqualo:=RANDOM(ProlificazioneSquali+1);
    Mondo[x,y].SqualoMosso:=FALSE;
    Mondo[x,y].FameSqualo:=0;
    GOTOXY(x+1,y+1); WRITE('o');
   END;
  DatiPesci[0]:=NumPesciIniziali;
  DatiSquali[0]:=NumSqualiIniziali;
  Tempo:=1;
  GOTOXY(1,1);
  WRITE(Tempo,': ',NumPesciIniziali,'P & ',
        NumSqualiIniziali,'S');
  {
   Ciclo principale (simulazione discreta del tempo)
  }
  WHILE TRUE DO
   BEGIN
```

```
       FOR x:=0 TO Xmax DO
      FOR y:=0 TO Ymax DO
       CASE Mondo[x,y].Cella OF
        Pesce: Mondo[x,y].PesceMosso:=FALSE;
        Squalo: Mondo[x,y].SqualoMosso:=FALSE
       END{CASE};
     {
      Ciclo di movimento casuale e di
      riproduzione dei pesci
     }
     FOR x:=0 TO Xmax DO
      FOR y:=0 TO Ymax DO
       BEGIN
        IF (Mondo[x,y].Cella=Pesce) AND
           (Mondo[x,y].PesceMosso=FALSE)
        THEN
         BEGIN

Mondo[x,y].EtaPesce:=Mondo[x,y].EtaPesce+1;
          NumCelleLibere:=0;
          IF Mondo[DecX(x),y].Cella = Nulla
THEN
           BEGIN
            CelleLibere[W]:=TRUE;
            NumCelleLibere:=NumCelleLibere+1
           END
          ELSE CelleLibere[W]:=FALSE;
          IF Mondo[DecX(x),IncY(y)].Cella =
Nulla THEN
           BEGIN
            CelleLibere[NW]:=TRUE;
            NumCelleLibere:=NumCelleLibere+1
           END
          ELSE CelleLibere[NW]:=FALSE;
          IF Mondo[x,IncY(y)].Cella = Nulla
THEN
           BEGIN
            CelleLibere[N]:=TRUE;
            NumCelleLibere:=NumCelleLibere+1
           END
          ELSE CelleLibere[N]:=FALSE;
          IF Mondo[IncX(x),IncY(y)].Cella =
Nulla THEN
           BEGIN
            CelleLibere[NE]:=TRUE;
            NumCelleLibere:=NumCelleLibere+1
           END
          ELSE CelleLibere[NE]:=FALSE;
          IF Mondo[IncX(x),y].Cella = Nulla
THEN
           BEGIN
            CelleLibere[E]:=TRUE;
            NumCelleLibere:=NumCelleLibere+1
           END
          ELSE CelleLibere[E]:=FALSE;
          IF Mondo[DecX(x),IncY(y)].Cella =
Nulla THEN
           BEGIN
            CelleLibere[SE]:=TRUE;
            NumCelleLibere:=NumCelleLibere+1
           END
          ELSE CelleLibere[SE]:=FALSE;
          IF Mondo[x,DecY(y)].Cella = Nulla
THEN
           BEGIN
            CelleLibere[S]:=TRUE;
            NumCelleLibere:=NumCelleLibere+1
           END
          ELSE CelleLibere[S]:=FALSE;
          IF Mondo[DecX(x),DecY(y)].Cella =
Nulla THEN
           BEGIN
            CelleLibere[SW]:=TRUE;
            NumCelleLibere:=NumCelleLibere+1
           END
          ELSE CelleLibere[SW]:=FALSE;
          IF NumCelleLibere>0 THEN
           BEGIN
            Scelta:=RANDOM(NumCelleLibere)+1;
            IF CelleLibere[W] THEN Conta:=1
ELSE Conta:=0;
            Pos:=W;
            WHILE Conta<Scelta DO
             BEGIN
              Pos:=SUCC(Pos);
              IF CelleLibere[Pos] THEN
Conta:=Conta+1
             END;
            CASE Pos OF
             W: BEGIN
                  newX:=DecX(x); newY:=y
                END;
             NW: BEGIN
                   newX:=DecX(x); newY:=IncY(y)
                 END;
             N: BEGIN
                  newX:=x; newY:=IncY(y)
                END;
             NE: BEGIN
                   newX:=IncX(x); newY:=IncY(y)
                 END;
             E: BEGIN
                  newX:=IncX(x); newY:=y
                END;
             SE: BEGIN
                   newX:=IncX(x); newY:=DecY(y)
                 END;
             S: BEGIN
                  newX:=x; newY:=DecY(y)
                END;
             SW: BEGIN
                   newX:=DecX(x); newY:=DecY(y)
                 END
            END{CASE};
            Mondo[newX,newY].Cella:=Pesce;

Mondo[newX,newY].EtaPesce:=Mondo[x,y].EtaPesce;
            Mondo[newX,newY].PesceMosso:=TRUE;
            IF
Mondo[x,y].EtaPesce=ProlificazionePesci THEN
             BEGIN
              Mondo[x,y].PesceMosso:=TRUE;
              Mondo[x,y].EtaPesce:=0;
              Mondo[newX,newY].EtaPesce:=0
             END
            ELSE Mondo[x,y].Cella:=Nulla
           END
         END
       END;
     {
      Ciclo di predazione o di movimento
      casuale, di morte per inedia e di
      riproduzione degli squali
     }
     FOR x:=0 TO Xmax DO
      FOR y:=0 TO Ymax DO
```

```
         BEGIN
        IF (Mondo[x,y].Cella=Squalo) AND
           (Mondo[x,y].SqualoMosso=FALSE)
        THEN
         BEGIN

Mondo[x,y].EtaSqualo:=Mondo[x,y].EtaSqualo+1;
          NumCellePesce:=0;
          IF Mondo[DecX(x),y].Cella = Pesce
THEN
           BEGIN
            CellePesce[W]:=TRUE;
            NumCellePesce:=NumCellePesce+1
           END
          ELSE CellePesce[W]:=FALSE;
          IF Mondo[DecX(x),IncY(y)].Cella =
Pesce THEN
           BEGIN
            CellePesce[NW]:=TRUE;
            NumCellePesce:=NumCellePesce+1
           END
          ELSE CellePesce[NW]:=FALSE;
          IF Mondo[x,IncY(y)].Cella = Pesce
THEN
           BEGIN
            CellePesce[N]:=TRUE;
            NumCellePesce:=NumCellePesce+1
           END
          ELSE CellePesce[N]:=FALSE;
          IF Mondo[IncX(x),IncY(y)].Cella =
Pesce THEN
           BEGIN
            CellePesce[NE]:=TRUE;
            NumCellePesce:=NumCellePesce+1
           END
          ELSE CellePesce[NE]:=FALSE;
          IF Mondo[IncX(x),y].Cella = Pesce
THEN
           BEGIN
            CellePesce[E]:=TRUE;
            NumCellePesce:=NumCellePesce+1
           END
          ELSE CellePesce[E]:=FALSE;
          IF Mondo[DecX(x),IncY(y)].Cella =
Pesce THEN
           BEGIN
            CellePesce[SE]:=TRUE;
            NumCellePesce:=NumCellePesce+1
           END
          ELSE CellePesce[SE]:=FALSE;
          IF Mondo[x,DecY(y)].Cella = Pesce
THEN
           BEGIN
            CellePesce[S]:=TRUE;
            NumCellePesce:=NumCellePesce+1
           END
          ELSE CellePesce[S]:=FALSE;
          IF Mondo[DecX(x),DecY(y)].Cella =
Pesce THEN
           BEGIN
            CellePesce[SW]:=TRUE;
            NumCellePesce:=NumCellePesce+1
           END
          ELSE CellePesce[SW]:=FALSE;
          IF NumCellePesce>0 THEN
           BEGIN
            Scelta:=RANDOM(NumCellePesce)+1;
            IF CellePesce[W] THEN Conta:=1 ELSE
Conta:=0;
            Pos:=W;
            WHILE Conta<Scelta DO
             BEGIN
              Pos:=SUCC(Pos);
              IF CellePesce[Pos] THEN
Conta:=Conta+1
             END;
            CASE Pos OF
             W: BEGIN
                 newX:=DecX(x); newY:=y
                END;
             NW: BEGIN
                  newX:=DecX(x); newY:=IncY(y)
                 END;
             N: BEGIN
                 newX:=x; newY:=IncY(y)
                END;
             NE: BEGIN
                  newX:=IncX(x); newY:=IncY(y)
                 END;
             E: BEGIN
                 newX:=IncX(x); newY:=y
                END;
             SE: BEGIN
                  newX:=IncX(x); newY:=DecY(y)
                 END;
             S: BEGIN
                 newX:=x; newY:=DecY(y)
                END;
             SW: BEGIN
                  newX:=DecX(x); newY:=DecY(y)
                 END
            END{CASE};
            Mondo[newX,newY].Cella:=Squalo;
            Mondo[newX,newY].EtaSqualo:=
             Mondo[x,y].EtaSqualo;
            Mondo[newX,newY].SqualoMosso:=TRUE;
            Mondo[newX,newY].FameSqualo:=0;
            IF
Mondo[x,y].EtaSqualo=ProlificazioneSquali
            THEN
             BEGIN
              Mondo[x,y].SqualoMosso:=TRUE;
              Mondo[x,y].EtaSqualo:=0;
              Mondo[newX,newY].EtaSqualo:=0;
              Mondo[x,y].FameSqualo:=0
             END
            ELSE Mondo[x,y].Cella:=Nulla
           END
          ELSE
           IF Mondo[x,y].FameSqualo>InediaSquali
THEN
            Mondo[x,y].Cella:=Nulla
           ELSE
            BEGIN
             NumCelleLibere:=0;
             IF Mondo[DecX(x),y].Cella = Nulla
THEN
              BEGIN
               CelleLibere[W]:=TRUE;
               NumCelleLibere:=NumCelleLibere+1
              END
             ELSE CelleLibere[W]:=FALSE;
             IF Mondo[DecX(x),IncY(y)].Cella =
Nulla THEN
              BEGIN
               CelleLibere[NW]:=TRUE;
```

```
NumCelleLibere:=NumCelleLibere+1
              END
             ELSE CelleLibere[NW]:=FALSE;
             IF Mondo[x,IncY(y)].Cella = Nulla
THEN
              BEGIN
               CelleLibere[N]:=TRUE;
               NumCelleLibere:=NumCelleLibere+1
             END
             ELSE CelleLibere[N]:=FALSE;
             IF Mondo[IncX(x),IncY(y)].Cella =
Nulla THEN
              BEGIN
               CelleLibere[NE]:=TRUE;
               NumCelleLibere:=NumCelleLibere+1
             END
             ELSE CelleLibere[NE]:=FALSE;
             IF Mondo[IncX(x),y].Cella = Nulla
THEN
              BEGIN
               CelleLibere[E]:=TRUE;
               NumCelleLibere:=NumCelleLibere+1
              END
             ELSE CelleLibere[E]:=FALSE;
             IF Mondo[DecX(x),IncY(y)].Cella =
Nulla THEN
              BEGIN
               CelleLibere[SE]:=TRUE;
               NumCelleLibere:=NumCelleLibere+1
              END
             ELSE CelleLibere[SE]:=FALSE;
             IF Mondo[x,DecY(y)].Cella = Nulla
THEN
              BEGIN
               CelleLibere[S]:=TRUE;
               NumCelleLibere:=NumCelleLibere+1
              END
             ELSE CelleLibere[S]:=FALSE;
             IF Mondo[DecX(x),DecY(y)].Cella =
Nulla THEN
              BEGIN
               CelleLibere[SW]:=TRUE;
               NumCelleLibere:=NumCelleLibere+1
              END
             ELSE CelleLibere[SW]:=FALSE;
             IF NumCelleLibere>0 THEN
              BEGIN

Scelta:=RANDOM(NumCelleLibere)+1;
               IF CelleLibere[W]
               THEN Conta:=1
               ELSE Conta:=0;
               Pos:=W;
               WHILE Conta<Scelta DO
                BEGIN
                 Pos:=SUCC(Pos);
                 IF CelleLibere[Pos] THEN
Conta:=Conta+1
                END;
               CASE Pos OF
                W: BEGIN
                     newX:=DecX(x); newY:=y
                   END;
                NW: BEGIN
                      newX:=DecX(x);
newY:=IncY(y)
                    END;
                N: BEGIN
                     newX:=x; newY:=IncY(y)
                   END;
                NE: BEGIN
                      newX:=IncX(x);
newY:=IncY(y)
                    END;
                E: BEGIN
                     newX:=IncX(x); newY:=y
                   END;
                SE: BEGIN
                      newX:=IncX(x);
newY:=DecY(y)
                    END;
                S: BEGIN
                     newX:=x; newY:=DecY(y)
                   END;
                SW: BEGIN
                      newX:=DecX(x);
newY:=DecY(y)
                    END
               END{CASE};
               Mondo[newX,newY].Cella:=Squalo;
               Mondo[newX,newY].EtaSqualo:=
                Mondo[x,y].EtaSqualo;
               Mondo[newX,newY].FameSqualo:=
                Mondo[x,y].FameSqualo+1;

Mondo[newX,newY].SqualoMosso:=TRUE;
               IF
Mondo[x,y].EtaSqualo=ProlificazioneSquali
               THEN
                BEGIN
                 Mondo[x,y].SqualoMosso:=TRUE;
                 Mondo[x,y].EtaSqualo:=0;
                 Mondo[newX,newY].EtaSqualo:=0;
                 Mondo[x,y].FameSqualo:=0
                END
               ELSE Mondo[x,y].Cella:=Nulla
              END
            END
          END
        END;
    {
     Visualizzazione e rilevazione dati
    }
    CLRSCR;
    ContaPesci:=0;
    ContaSquali:=0;
    FOR x:=0 TO Xmax DO
     FOR y:=0 TO Ymax DO
      BEGIN
       CASE Mondo[x,y].Cella OF
        Pesce: BEGIN
                 GOTOXY(x+1,y+1);
                 WRITE('.');
                 ContaPesci:=ContaPesci+1
               END;
        Squalo: BEGIN
                  GOTOXY(x+1,y+1);
                  WRITE('o');
                  ContaSquali:=ContaSquali+1
                END
       END{CASE}
      END;
    GOTOXY(1,1);
    WRITE(Tempo,': ',ContaPesci,'P &
',ContaSquali,'S');
```

```
      DatiPesci[Tempo]:=ContaPesci;
    DatiSquali[Tempo]:=ContaSquali;
    IF Tempo=TempoEsperimento THEN GOTO Fine;
    Tempo:=Tempo+1
   END;
 Fine:
  {
   Memorizzazione su file dei dati
  }
  REWRITE(FileDati);
  WRITE(FileDati,0);
  FOR i:=1 TO TempoEsperimento DO
 WRITE(FileDati,';',i);
  WRITELN(FileDati);
  WRITE(FileDati,DatiPesci[0]);
  FOR i:=1 TO TempoEsperimento DO
   WRITE(FileDati,';',DatiPesci[i]);
  WRITELN(FileDati);
  WRITE(FileDati,DatiSquali[0]*10);
   { coefficiente moltiplicativo di correzione
}
  FOR i:=1 TO TempoEsperimento DO
   WRITE(FileDati,';',DatiSquali[i]*10);
  CLOSE(FileDati)
 END.
```

LISTATO 2

```
  > with(DEtools);

> with(plots);


> sol:=dsolve({diff(x(t),t)=.25*x(t)-
 .5*x(t)*y(t),
                diff(y(t),t)=-
.125*y(t)+.5*x(t)*y(t),
             x(0)=.5,y(0)=.1},{x(t),y(t)},
            type=numeric,output=listprocedure);

> prey:=subs(sol,x(t));

> predator:=subs(sol,y(t));

> plot({prey,predator},0..125);

> plot([prey,predator,0..125],0..1.5,0..1.75);

> odeplot(sol,[t,x(t),y(t)],0..125,
  numpoints=1000);

> DEplot2([.25*x-.5*x*y,-.125*y+.5*x*y],
  [x,y],0..125,

{[0,.5,.1],[0,.5,.5],[0,.1,.5]},x=0..2,y=0..2,
           stepsize=1);
```

```
> sol:=dsolve({diff(x(t),t)=
  .25*x(t)-.125*x(t)^2-.5*x(t)*y(t),
                diff(y(t),t)=-
.125*y(t)+.5*x(t)*y(t),x(0)=.5,
             y(0)=.1},{x(t),y(t)},
            type=numeric,output=listprocedure);

> prey:=subs(sol,x(t));

> predator:=subs(sol,y(t));

> plot({prey,predator},0..250);

> plot([prey,predator,0..250],0..1,0..1);

> odeplot(sol,[t,x(t),y(t)],0..250,
  numpoints=1000);

> DEplot2([.25*x-.125*x^2-.5*x*y,
  -.125*y+.5*x*y],[x,y],

0..250,{[0,.5,.1],[0,.5,.5],[0,.1,.5]},
          x=0..2,y=0..2,stepsize=1);


> DEplot([10*(y-x),28*x-y-x*z,
  -(8/3)*z+x*y],[x,y,z],t=0..50,
          {[0,5,5,5]},stepsize=.05);
```

Le equazioni differenziali

La soluzione di una equazione, se esiste, è un numero o un insieme di numeri; per esempio la soluzione dell'equazione $x^2-1=0$ è data dall'insieme $X=\{x / x=-1,x=1\}$. La soluzione di una equazione differenziale è invece, quando esiste, una funzione o un insieme di funzioni. La seguente è una delle più semplici equazioni differenziali (la notazione y'(x) per la derivata è formalmente equivalente a dy/dx):

$y'(x) = x$

Si tratta di determinare la funzione y(x) la cui derivata prima è y'(x)=x ed è sufficiente calcolare un integrale banale per trovare $y(x)=x^2/2+k$. L'applicazione dell'operatore integrale produce come risultato un insieme di funzioni che differiscono tra loro per il valore di un parametro reale additivo k: il risultato di una equazione differenziale è spesso definito da una condizione iniziale $y(x_0)=y_0$; per esempio nel caso precedente la condizione iniziale y(0)=1 comporterebbe $1=0^2+k$ (per sostituzione nel risultato dell'integrazione) e di conseguenza k=1. In definitiva il risultato dell'equazione differenziale y'(x)=x con condizione iniziale y(0)=1 è la funzione $y(x)=x^2/2+1$ (una parabola nel piano cartesiano). Non tutte le equazioni differenziali si risolvono così semplicemente, ma ricorrendo alle funzionalità di calcolo simbolico di programmi come *Mathematica* della Wolfram Research o *MapleV* della Waterloo Maple Software è possibile risolvere equazioni differenziali anche complesse senza conoscere i dettagli delle tecniche risolutive. Per esempio, volendo risolvere l'equazione differenziale

$x'(t) = rx(1-x/k)$ con condizione iniziale $y(0)=x_0$, r e k parametri reali

è sufficiente impostare il comando di *Mathematica*

```
DSolve[{x'[t]==r*x[t]*(1-x[t]/k),x[0]==x0},
       x[t],t]
```

per ottenere come risultato

```
{{x[t]->Ertk/(-1+Ert+k/x0)}}
```

cioè la funzione $x(t)=ke^{rt}/(e^{rt}+k/x_0-1)=k/(1-(1-k/x_0)e^{-rt})$.

Le funzioni derivate hanno una interpretazione fisica di importanza fondamentale nei modelli matematici dei sistemi dinamici: la derivata prima rappresenta infatti la variazione di una funzione e la derivata seconda (la derivata della funzione derivata prima) rappresenta la velocità di variazione della funzione stessa. Se ad esempio **s(t)** è la funzione che rappresenta lo spostamento nel tempo di un corpo lungo una retta, allora la derivata prima **s'(t)** rappresenta l'andamento nel tempo della velocità del corpo e la derivata seconda **s''(t)** rappresenta l'accelerazione a cui è sottoposto il corpo in un dato istante **t**. Fatta questa osservazione è immediato descrivere il comportamento di un sistema meccanico (un esempio particolare di sistema dinamico) per mezzo di una equazione differenziale. Volendo per esempio studiare il moto di caduta libera di un corpo di massa **m** da una altezza **h** senza trascurare l'attrito dell'aria:

- l'equazione fondamentale della dinamica (equazione di Newton) è **F(t)=m•a(t)** (la somma vettoriale delle forze cui è sottoposto il corpo è uguale, istante per istante, al prodotto tra massa ed accelerazione del corpo stesso),
- la forza di attrazione gravitazionale della Terra è costante nel tempo ed è data da $\mathbf{F_g=-(m•g)}$ (dove **g** è l'accelerazione di gravità che vale circa 9,8 m/s^2),
- la forza di attrito viscoso dovuta alla presenza dell'aria è proporzionale, con coefficiente moltiplicativo **k**, alla velocità del corpo: $\mathbf{F_a=k•v(t)}$;

  ricordando che **a(t)=s''(t)** e che **v(t)=s'(t)** si ricava che:

- **(m•g) + k•s'(t) = m•s''(t)** con condizioni iniziali **s(0)=h** e **v(0)=s'(0)=0** (la velocità iniziale del corpo è infatti nulla).

In *Mathematica*:

```
DSolve[{-(m*g)+k*s'[t]==m*s''[t],s[0]
= h,s'[0]=0}, s[t],t]

{{s[t]->(hk2+gm2-E(kt)/mgm2+gkmt)/k2}}
```

Si noti che la soluzione simbolica è stata ottenuta senza specificare il valore dei parametri: per tracciare un grafico della soluzione è ovviamente necessario assegnare un valore numerico a **g**, **h**, **m** e **k**. Un ottimo testo per chi intenda affrontare lo studio delle equazioni differenziali con *Mathematica* e che comprende una sintetica introduzione ad un uso più generale del programma è: K. Coombes, B. Hurt, R. Lipsman, J. Osborn, G. Stuck, "Differential Equations with *Mathematica*", Wiley, 1995; è in preparazione la versione per *MapleV* del medesimo testo.

Integrazione numerica di equazioni differenziali

Anche nel caso di una semplice equazione come *cos* x = x può essere impossibile determinarne la soluzione per via analitica utilizzando i metodi del calcolo simbolico. Tuttavia una soluzione esiste ed è possibile calcolarla con l'approssimazione desiderata applicando un metodo numerico. In questo caso particolare è possibile ricorrere al cosiddetto algoritmo iterativo che, a partire da un valore iniziale x_0 scelto casualmente, calcola una successione ricorsiva di valori x_i = *cos* x_{i-1} avente come limite la soluzione dell'equazione:

$x_0 = 0.5$
$x_1 = \cos x_0 = 0.87...$
$x_2 = \cos x_1 = 0.63...$
$x_3 = \cos x_2 = 0.80...$
$x_4 = \cos x_3 = 0.69...$
$x_5 = \cos x_4 = 0.76...$
$x_6 = \cos x_5 = 0.71...$
$x_7 = \cos x_6 = 0.75...$
$x_8 = \cos x_7 = 0.73...$
$x_9 = \cos x_8 = 0.74...$

Nel caso di una equazione differenziale la non esistenza di una tecnica simbolica di risoluzione è spesso aggravata dalla possibilità che esista una soluzione non esprimibile in termini di funzioni elementari. Per esempio l'equazione differenziale

$$\frac{dy}{dx} = e^{-x^2}$$

ha per soluzione

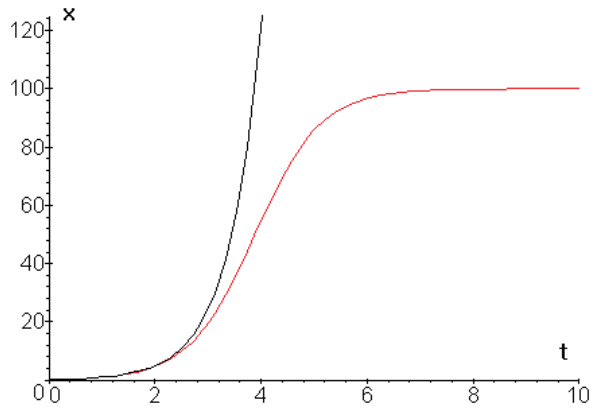
$$y(x) = \int e^{-x^2}\,dx + k$$

ma l'integrale indefinito che vi compare, pur rappresentando una funzione, non è esprimibile in termini di funzioni più elementari e non può, di conseguenza, essere eliminato. Se però aggiungiamo la condizione iniziale $\mathbf{y(x_0)=y_0}$ la soluzione è data da

$$y(x) = y_0 + \int_{x_0}^{x} e^{-t^2}\,dt$$

e l'integrale definito che vi compare può essere approssimato - per ogni valore di x - con un metodo di integrazione numerica. All'impossibilità di ottenere una espressione analitica esatta per y(x) si ovvia con una procedura che calcoli un valore approssimato y a partire da un valore x fornito come parametro: questa soluzione si dimostra essere soddisfacente per molti scopi pratici. Nella figura è riportato in rosso il grafico della soluzione numerica calcolata con *MapleV* impostando i seguenti comandi (la notazione diff(y(x),x) è formalmente equivalente a y'(x) ed a dy/dx):

```
gauss:=proc(x) exp(-x^2) end;
sol:=dsolve({diff(y(x),x)
=exp(-x^2),y(0)=1},y(x),
```



Il grafico della funzione exp(-x^2) e della sua funzione integrale calcolata con un metodo numerico.