a cura di Pierluigi Panunzi

Con il programma che presentiamo in questo numero ci addentreremo in un campo alquanto difficile da applicare alle calcolatrici della Texas: il plottaggio di funzioni.

Come tutti i lettori texasiani ben sanno, le caratteristiche grafiche della loro macchinetta sono ben lungi dall'essere considerate il non plus ultra: quel poco che si riesce ad ottenere risulta sempre il frutto di lunghissimi programmi tra l'altro con durate di elaborazione chilometriche...

Comunque bisogna ricordare che le calcolatrici in questione sono oramai alquanto anzianotte soprattutto se viste nell'ottica odierna, tanto siamo abituati a vedere sfornare quasi ogni giorno dei mirabili prodigi dell'elettronica, capaci di fare anche l'impossibile.

Non vogliamo certo denigrare i "nostri" gioielli di marca TI in quanto, tutto sommato, le loro funzioni le svolgono egregiamente.

Ritornando al nostro programma, i lettori ritroveranno una vecchia conoscenza, ad ulteriore riprova che dei programmi pubblicati viene saggiata innanzitutto la qualità e solo in ultima analisi "la provenienza".

Graficomania

(o apoteosi della grafica "texana")

di Andrea Cantadori (Parma)

Il problema

Supponiamo data una funzione di due variabili, x ed y, "sufficientemente" regolare, nella forma f(x,y) = 0 e cioè in forma implicita.

È noto ai possessori delle TI che, per riprodurre il grafico di una tale funzione su stampante, occorre dapprima esplicitarla (il che risulta alcune volte molto difficile se non impossibile) cioè trasformarla in modo da avere direttamente il valore della y al variare della x.

Inutile dire che questa necessità, dovuta al fatto che la carta della stampante può solo muoversi in un senso, diminuisce notevolmente le "capacità grafiche" (se così vogliamo chiamarle) della nostra PC 100C.

Mi sono quindi scervellato ed ho ottenuto un risultato interessante, come vedremo.

Infatti questo programma consente di:

1) Disegnare grafici di funzioni del tipo f(x,y) = 0: questo caso comprende tra l'altro anche le funzioni del tipo y = f(x), giacché basta scriverle come y-f(x) = 0 e considerare come funzione da disegnare la F(x,y) = y-f(x) = 0.

2) Tarare automaticamente le scale grafiche, senza bisogno quindi di impostare "strani" coefficienti.

3) Disegnare, pure automaticamente, gli assi x e y cartesiani, sempreché cadano nell'intervallo da noi scelto.

4) Disegnare anche più funzioni contemporaneamente. Ad esempio se vogliamo disegnare i grafici di f(x,y) = 0 e g(x,y) = 0, basterà disegnare il grafico della funzione F (x,y) = f (x,y) g (x,y) = 0.

5) Come eventualità, variando l'intervallo di definizione, di ottenere delle "finestre grafiche", con ingrandimento di particolari.

Il programma ovviamente ha dei limiti, tra i quali la lentezza. Ma voglio precisare che la mia intenzione non è già quella di risolvere il problema compiutamente, quanto quella di dare un primo contributo allo sviluppo di un software grafico per la TI, per il quale spero parteciperanno altri lettori con loro idee. Ad ogni modo, prima di andare a commentare il procedimento seguito, date un'occhiata alla figura 1.

Si tratta della "lemniscata di Bernoulli", di equazione:

$x^2 + y^2 + 1 - \sqrt{4x^2 + 1} = 0$

disegnata nell'intervallo $-1.5 \leqslant x \leqslant 1.5$ $-0.5 \leqslant y \leqslant 0.5$.

La risoluzione è in questo caso di 100 × 25, cioè 2.500 punti. Spero che l'esempio visivo vi abbia invogliato a leggere anche il resto.

L'impostazione

Immaginiamo di avere uno schermo video, invece della stampante, a disposizione. Supponiamo poi di voler disegnare su di esso una funzione esplicita y = f(x).

Si potrebbe allora, in via teorica, definire la risoluzione dello schermo, cioè dividerlo in tanti rettangolini (i "pixel") disposti su file orizzontali e verticali in modo da ricoprire tutto lo schermo. Partendo poi dal pixel in alto a sinistra, si potrebbe vedere se la f(x) fornisce un valore di y contennuto in esso o no ed in caso affermativo si dovrebbe "accendere" il pixel stesso. Proseguendo così, con una scansione successiva per linee orizzontali, si arriverebbe a definire il diagramma di f(x) per punti.

Facciamo un passo più in là considerando ora una funzione implicita f(x,y) = 0. È evidente che in questo caso il metodo di prima non è più applicabile direttamente. Pur tuttavia, immaginiamo di scansionare ancora lo schermo per linee orizzontali e, per ogni pixel, prendiamo il suo centro, vale a dire (vedi fig. 2) il punto C di coordinate

$(x_1 + x_2)/ 2$ e $(y_1 + y_2)/ 2$

Calcoliamo quindi la f(x,y) in quel punto: è evidente che assai di rado ci potrà risultare 0, per cui solo pochissimi pixel verrebbero accesi, tanto da rendere inutilizzabile il disegno ottenuto.

Abbandoniamo allora le nostre aspirazioni ad una "precisione" del grafico molto elevata ed accontentiamoci di una approssimazione: in altre parole, accenderemo il pixel non già quando è esattamente f(x,y) = 0, ma quando, più in generale, è $f(x,y) \leqslant h$, avendo indicato con "h" un'opportuna quantità da definirsi. È evidente che, scelto "il miglior" h, avremo una buona descrizione dell'andamento della funzione.

Tuttavia, riferendoci alla già citata figura 1 appaiono ben evidenti i limiti di questo procedimento: eccesso di pixel accesi in una parte ed eventualmente lacune in un'altra.

D'altra parte le mie difficoltà maggiori riguardano proprio la scelta del valore di "h": darò poi il criterio empirico da me trovato, ma se qualcuno è in grado di trovare un procedimento per individuare l'"h" migliore si faccia sentire!

Vediamo ora come il procedimento descritto è stato applicato alla mia TI 59.

Il programma

I dati necessari per il programma sono, oltre la f(x,y) stessa:

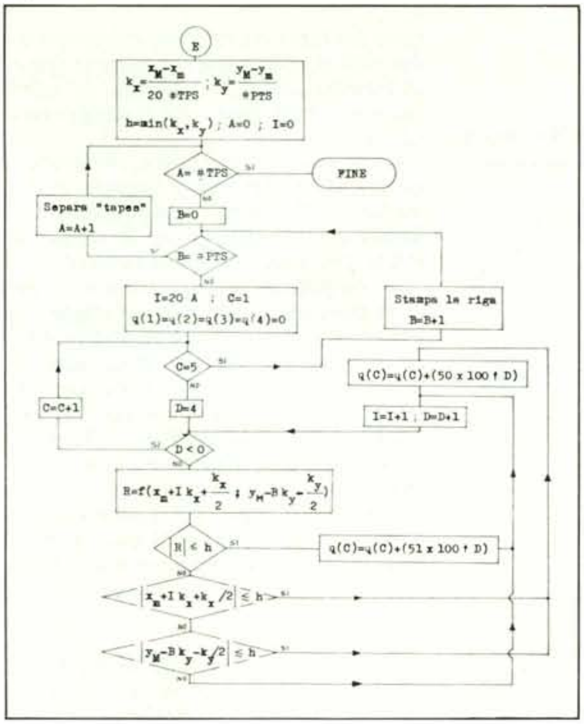
1) limiti massimo e minimo di variabilità della x, cioè x_M e x_m rispettivamente

Graficomania

```
PLOTTAGGIO DI
FUNZIONI
DEL TIPO
F(X,Y)=0

000  76 LBL
001  11  A
002  42 STO
003  13  13
004  92 RTN
005  76 LBL
006  16  A'
007  42 STO
008  11  11
009  92 RTN
010  76 LBL
011  12  B
012  42 STO
013  12  12
014  92 RTN
015  76 LBL
016  17  B'
017  42 STO
018  14  14
019  92 RTN
020  76 LBL
021  13  C
022  42 STO
023  09  09
024  92 RTN
025  76 LBL
026  14  D
027  42 STO
028  10  10
029  92 RTN
030  76 LBL
031  15  E
032  43 RCL
033  11  11
034  75  -
035  43 RCL
036  13  13
037  95  =
038  55  ÷
039  43 RCL
040  09  09
041  55  ÷
042  02  2
043  00  0
044  95  =
045  42 STO
046  11  11
047  43 RCL
048  14  14
049  75  -
050  43 RCL
051  12  12
052  95  =
053  55  ÷
054  43 RCL
055  10  10
056  95  =
057  42 STO
058  12  12
059  32 X:T
060  43 RCL
061  11  11
062  77  GE
063  00  00
064  70  70
065  42 STO
066  17  17
067  61 GTO
068  00  00
069  74  74
070  43 RCL
071  12  12
072  42 STO
073  17  17
074  25 CLR
075  42 STO
076  05  05
077  42 STO
078  00  00
079  43 RCL
080  09  09
081  32 X:T
082  43 RCL
083  05  05
084  67  EQ
085  02  02
086  67  67
087  25 CLR
088  42 STO
089  06  06
090  43 RCL
091  10  10
092  32 X:T
093  43 RCL
094  06  06
095  67  EQ
096  02  02
097  17  17
098  02  2
099  00  0
100  65  ×
101  43 RCL
102  05  05
103  95  =
104  42 STO
105  00  00
106  01  1
107  42 STO
108  07  07
109  05  5
110  32 X:T
111  43 RCL
112  07  07
113  67  EQ
114  02  02
115  47  47
116  25 CLR
117  72 ST*
118  07  07
119  04  4
120  42 STO
121  08  08
122  01  1
123  94 +/-
124  32 X:T
125  43 RCL
126  08  08
127  67  EQ
128  02  02
129  54  54
130  43 RCL
131  13  13
132  85  +
133  43 RCL
134  00  00
135  65  ×
136  43 RCL
137  11  11
138  85  +
139  43 RCL
140  11  11
141  55  ÷
142  02  2
143  95  =
144  42 STO
145  15  15
146  43 RCL
147  14  14
148  75  -
149  43 RCL
150  06  06
151  65  ×
152  43 RCL
153  12  12
154  75  -
155  43 RCL
156  12  12
157  55  ÷
158  02  2
159  95  =
160  42 STO
161  16  16
162  71 SBR
163  02  02
164  71  71
165  50 I×I
166  32 X:T
167  43 RCL
168  17  17
169  77  GE
170  02  02
171  02  02
172  43 RCL
173  15  15
174  50 I×I
175  32 X:T
176  43 RCL
177  17  17
178  77  GE
179  01  01
180  97  97
181  43 RCL
182  16  16
183  50 I×I
184  32 X:T
185  43 RCL
186  17  17
187  77  GE
188  01  01
189  97  97
190  69  OP
191  20  20
192  69  OP
193  38  38
194  61 GTO
195  01  01
196  22  22
197  05  5
198  00  0
199  61 GTO
200  02  02
201  04  04
202  05  5
203  01  1
204  65  ×
205  01  1
206  00  0
207  00  0
208  45  Y×
209  43 RCL
210  08  08
211  95  =
212  74 SM*
213  07  07
214  61 GTO
215  01  01
216  90  90
217  98 ADV
218  06  6
219  04  4
220  06  6
221  04  4
222  06  6
223  04  4
224  06  6
225  04  4
226  06  6
227  04  4
228  69  OP
229  01  01
230  69  OP
231  02  02
232  69  OP
233  03  03
234  69  OP
235  04  04
236  69  OP
237  05  05
238  98 ADV
239  98 ADV
240  98 ADV
241  98 ADV
242  69  OP
243  25  25
244  61 GTO
245  00  00
246  79  79
247  69  OP
248  05  05
249  69  OP
250  26  26
251  61 GTO
252  00  00
253  90  90
254  73 RC*
255  07  07
256  52  EE
257  22 INV
258  52  EE
259  84 OP*
260  07  07
261  69  OP
262  27  27
263  61 GTO
264  01  01
265  09  09
266  25 CLR
267  91 R/S
268  76 LBL
269  10  E'
270  31 LRN
271  00  0
272  00  0
273  00  0
274  00  0
275  00  0
276  00  0

001  11  A
006  16  A'
011  12  B
016  17  B'
021  13  C
026  14  D
031  15  E
269  10  E'
```



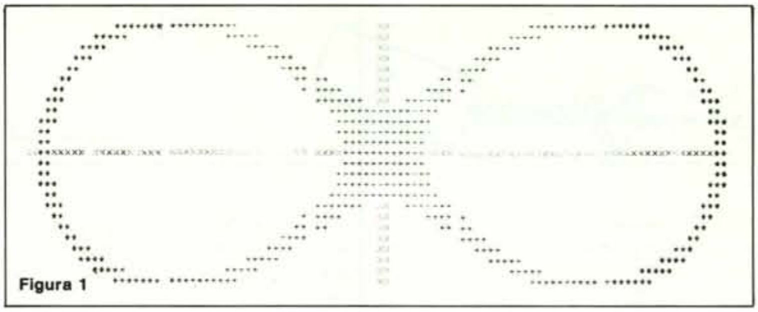


Figura 1

Sopra, esempio di plottaggio di funzione: si tratta della "Lemniscata di Bernoulli".

A sinistra, flow-chart del programma "Graficomania".

A destra, rappresentazione schematica delle coordinate di un "pixel".

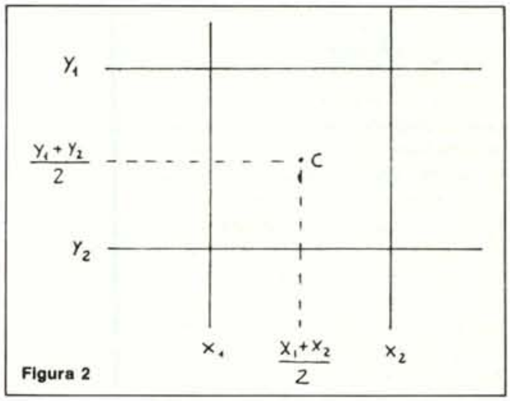


Figura 2

2) limiti analoghi per la variabile y, cioè y_M e y_m

3) il numero di tracce (tape), cioè il numero di fogli da accostare in senso orizzontale, vale a dire il numero di "segmenti" verticali (di "fette") in cui viene diviso il nostro video immaginario. Tale numero sarà indicato con ♯TPS e, poiché ogni "tape" contiene 20 punti per la x, è ovvio che la conoscenza di ♯TPS determina il numero di punti orizzontali

4) il numero di punti verticali (♯PTS). Ricordo che, per una corretta scala grafica, tale numero deve essere i 13/20 del numero di punti orizzontali. Questo è dovuto al fatto che 20 punti orizzontali occupano in lunghezza lo stesso spazio di 13 punti stampati in verticale. È comunque possibile inserire un qualunque altro valore, solo che così si avrà un allungamento o un accorciamento dell'immagine reale.

Ricordiamo ora il modo di scrivere messaggi con la PC 100C. Ogni striscia di carta è "divisa" in quattro settori, numerati da 1 a 4 da sinistra a destra, corrispondenti a 4 buffer di stampa ai quali si accede tramite le istruzioni Op 01, Op 02, Op 03 ed Op 04 e, per la stampa della linea, Op 05.

Siccome per avere il nostro "video" dovremo unire tante strisce (tape), la scansione non avverrà proprio come su uno schermo reale, ma in maniera molto più strutturata. Vediamo come.

Si parte dalla prima striscia, alla prima riga; si prende di questa il primo settore, che sarà formato da 5 pixel. Questi vengono scansionati da destra a sinistra, cioè in modo contrario a quanto avviene nel complesso.

Se la funzione è in valore assoluto minore di un certo "h", nel pixel viene posto il valore 51 (corrispondente all'asterisco), se invece c'è solo l'asse x o y viene posto 41 (corrispondente al simbolo "x"), altrimenti si lascia 0 (cioè un "blank"). Esauriti i 5 pixel del primo settore, si passa al secondo e poi si ripete il ragionamento fino al quarto.

Esaminati tutti e quattro i settori della prima riga e riempiti i relativi quattro buffer di stampa, la riga viene stampata. Si procede poi in modo del tutto analogo per la seconda linea e così via fino ad un numero di linee uguale a ♯PTS, cioè al numero di punti scelti in verticale.

Stampata un'intera striscia, viene stampata una riga di "=" di separazione e quindi si passa ad analizzare la seconda striscia e così via fino ad un numero di strisce maggiore di ♯TPS.

In definitiva la gerarchia tra i contatori utilizzati è la seguente:

A Contatore numero strisce (♯TPS)
B Contatore numero righe (♯PTS)
C Contatore del buffer di stampa
D Contatore posizione del pixel nel buffer

Per questi quattro contatori si hanno i seguenti range di valori

A	0 ... (♯TPS-1)	R05
B	0 ... (♯PTS-1)	R06
C	1 ... 4	R07
D	0 ... 4	R08

Questo per i contatori; nel flow-chart sono poi usate altre variabili il cui significato è di facile interpretazione.

In particolare sono utilizzate le memorie da 01 a 04 come "buffer dei buffer di stampa" e sono indicate nel flow-chart con "q ( )".

Mi resta ora solo da dire che prima viene eseguito il test per l'eventuale disegno della f(x,y) e solo qualora questo non dia la presenza della curva in un certo tratto si effettuerà il testo per il disegno degli assi.

Pertanto se gli assi e la funzione si intersecano, viene disegnata solo la funzione.

Inoltre, come si vedrà, i codici 51 e 50 (relativi ai simboli asterisco e crocetta) vengono posizionati correttamente all'interno del buffer moltiplicandoli per un valore 100 elevato a "D" (il contatore visto prima).

Per quanto riguarda poi il numero "h", il criterio da me trovato è semplice: viene posto uguale al minimo tra k_x e k_y, cioè tra gli incrementi delle variabili x ed y.

Naturalmente tutto ciò è approssimato rispetto ad una soluzione "esatta" del problema.

Devo comunque confessare che non ho trovato una soluzione migliore.

L'uso del programma

È molto semplice. Siccome, per ovvi motivi di tempi di elaborazione, si è preferito chiamare da programma la f(x,y) con una istruzione del tipo SBR "indirizzo" anziché SBR "etichetta", la funzione stessa andrà impostata sempre a partire dal passo 271. D'altro canto il programma, all'atto dell'inserimento di f(x,y) provvede da solo ad entrare in modo LRN al passo 271.

Vediamo ora l'esatta sequenza di operazioni da compiere.

1) Premere E': la calcolatrice entra in modo LRN. Impostare dunque la f(x,y) usando R15 per la variabile x ed R16 per la y. Ricordare le solite norme, cioè non usare i tasti = e CLR, fare uso di parentesi e terminare con INV SBR.

2) Impostare x_M e premere A'

3) Impostare x_m e premere A

4) Impostare y_M e premere B'

5) Impostare y_m e premere B

6) Impostare ♯TPS e premere C

7) Impostare ♯PTS e premere D

8) Qualora si sia introdotto un dato sbagliato, semplicemente basta ribatterlo; quando tutti i dati sono stati impostati...

9) Premere E per iniziare la stampa... e attendere...

Importante: i valori di ♯PTS e ♯TPS devono essere interi!

Osservazioni ed esempi

Una prima osservazione riguarda la velocità di elaborazione: sono presenti ben 7 salti condizionati, il che porta ad una certa perdita di tempo.

In media ho rilevato che per esaminare, ed eventualmente riempire, uno dei nostri "pseudo-pixel" occorrono circa 5 secondi. Questo fatto, riferito alla figura 1, che abbiamo detto essere composta da 2500 punti, porta a circa 3 ore ed un quarto il tempo richiesto per l'elaborazione, il che non è poco.

Non si tratta quindi di un programma velocissimo, ma la cosa è dovuta anche alla complessità del flow-chart.

Bene, a questo punto penso di avere detto tutto: mi auguro che altri lettori contribuiscano all'argomento, magari utilizzando lo stesso algoritmo, un po' più articolato, per una grafica tridimensionale. Chissà... MC