3.7. De klassiska polynomens ortogonalitetsegenskaper

I tabellen på nästa sida anges egenskaperna för några av de vanligaste ortogonala polynomen. Polynomen är normerade så , att ortogonalitetsvillkoret

$$\int_a^b w(x)p_n(x)p_m(x)dx = \begin{cases} 0 & \text{om } m \neq n \\ 1 & \text{om } m = n \end{cases}$$

är uppfyllt. De satisfierar dessutom *Rodrigues' formel* [1]:

$$\boxed{\mathcal{P}_n(x) = \frac{1}{a_n w(x)} \frac{d^n}{dx^n}\{w(x)[g(x)]^n\},}$$

om $\mathcal{P}_n$ betecknar $P_n, C_n^{(\nu)}, \ldots$ etc.

Mera information om ortogonala polynom kan man t.ex. finna i M. Abramowitz and I.A. Stegun: *Handbook of mathematical functions*, Dover Publ., 1965.

[1] efter Olinde Rodrigues, fransk bankman och matematiker (1795-1891)

Polynomet $p_n(x)$	(a,b)	$w(x)$	$g(x)$	a_n
$(n+\frac{1}{2})^{\frac{1}{2}} P_n(x)$ Legendre	$(-1,+1)$	1	$1-x^2$	$(-1)^n 2^n n!$
$2^\nu \Gamma(\nu) \left[\frac{n!(n+\nu)}{2\pi\Gamma(n+2\nu)}\right]^{\frac{1}{2}} C_n^\nu(x)$ Gegenbauer	$(-1,+1)$	$(1-x^2)^{\nu-\frac{1}{2}}$	$1-x^2$	$(-1)^n 2^n n! \frac{\Gamma(2\nu)\Gamma(\nu+n+\frac{1}{2})}{\Gamma(\nu+\frac{1}{2})\Gamma(n+2\nu)}$
$\left[\frac{n!\Gamma(\alpha+\beta+n+1)(\alpha+\beta+2n+1)}{2^{\alpha+\beta+1}\Gamma(\alpha+n+1)\Gamma(\beta+n+1)}\right]^{\frac{1}{2}} P_n^{(\alpha,\beta)}$ Jacobi	$(-1,+1)$	$(1-x)^\alpha(1+x)^\beta$	$1-x^2$	$(-1)^n 2^n n!$
$\left[\frac{2}{\pi(1+\delta_{n0})}\right]^{\frac{1}{2}} T_n(x)$ Tjebysjev	$(-1,+1)$	$(1-x^2)^{-\frac{1}{2}}$	$1-x^2$	$(-1)^n 2^n \frac{\Gamma(n+\frac{1}{2})}{\sqrt{\pi}}$
$\left(\sqrt{\pi}2^n n!\right)^{-\frac{1}{2}} H_n(x)$ Hermite	$(-\infty,+\infty)$	e^{-x^2}	1	$(-1)^n$
$\left[\frac{n!}{\Gamma(1+\alpha+n}\right]^{\frac{1}{2}} L_n^{(\alpha)}(x)$ Laguerre	$(0,+\infty)$	$x^\alpha e^{-x}$	x	$n!$

3.8. Approximation med rationella funktioner

En rationell funktion är en funktion, som kan uttryckas som kvoten av två polynom. En karaktäristisk egenskap för sådana funktioner är, att om man adderar en konstant till argumentet, eller multiplicerar det med en konstant, så är den fortsättningsvis en rationell funktion.

Rationella funktioner är lämpliga approximationsfunktioner isynnerhet för funktioner som har diskontinuiteter (godtyckligt stora eller godtyckligt små värden, som också kallas poler) inom det studerade intervallet, eftersom en rationell funktion också kan ha poler (nollställen i nämnaren).

Antag att vi vill anpassa ett antal givna datapunkter $(x_i, y_i), i = 1, 2, \ldots$ till en rationell funktion

$$R_N(x) = \frac{a_0 + a_1 x + a_2 x^2 + \ldots + a_n x^n}{1 + b_1 x + b_2 x^2 + \ldots + b_m x^m}, \quad (N = m + n)$$

dvs vi önskar approximera $y = f(x)$ med $R_N(x)$, och den konstanta termen i nämnaren har valts lika med 1.

Hur kan vi beräkna koefficienterna a_k, b_k? Antag, att $f(x)$ kan utvecklas i en Taylorserie inom intervallet $-1 \leq x \leq +1$:

$$f(x) = c_0 + c_1 x + c_2 x^2 + \ldots, \quad c_k = f^{(k)}(0)/k!, \quad k = 1, 2, \ldots$$

Koefficienterna i den rationella funktionen kan då bestämmas så, att f och R_N överensstämmer i x, och likaså deras derivator t.o.m. ordningen N. Detta leder till en Padé–approximation för funktionen f (uppkallad efter den franske matematikern Henri Padé, som skrev en artikel om approximation med rationella funktioner år 1892). Denna approximation är ofta överlägsen en trunkerad Taylorserie.

Om vi avbryter Taylorserien efter N termer så kan skillnaden mellan $f(x)$ och $R_N(x)$ uttryckas

$$f(x) - R_N(x) = c_0 + c_1 x + c_2 x^2 + \ldots + c_N x^N - \frac{a_0 + a_1 x + a_2 x^2 + \ldots + a_n x^n}{1 + b_1 x + b_2 x^2 + \ldots + b_m x^m}$$

$$= \frac{(c_0 + c_1 x + \ldots + c_N x^N)(1 + b_1 x + \ldots + b_m x^m) - (a_0 + a_1 x + \ldots + a_n x^n)}{1 + b_1 x + \ldots + b_m x^m}$$

Genom att sätta $f(x) = R_N(x)$ för $x = 0$, så får vi $c_0 - a_0 = 0$.

För att de första N derivatorna av $f(x)$ och $R_N(x)$ skall försvinna i punkten $x = 0$, så måste koefficienterna för alla potenser av x i täljaren försvinna. Vi får till slut ett ekvationssystem med $m+n+1$ ekvationer i $m + n + 1$ obekanta:

$$
\begin{aligned}
c_0 - a_0 &= 0 \\
c_1 + c_0 b_1 - a_1 &= 0 \\
c_2 + c_1 b_1 + c_0 b_2 - a_2 &= 0 \\
&\vdots \\
c_n + c_{n-1} b_1 + \ldots + c_0 b_n - a_n &= 0 \\
c_{n+1} + c_n b_1 + \ldots + c_{n-m+1} b_m &= 0 \\
&\vdots \\
c_{n+m} + c_{n+m-1} b_1 + \ldots + c_n b_m &= 0,
\end{aligned}
$$

där $c_k = 0$ om $k < 0$ och $b_l = 0$ om $l > m$.

Genom att lösa detta ekvationssystem, kan alla koefficienterna beräknas. En rätt bra Padé–approximation för exponentialfunktionen är t.ex.

$$R_{3+3}(x) = \frac{120 + 60x + 12x^2 + x^3}{120 - 60x + 12x^2 - x^3},$$

som ger värdet 2.71831 för e, då den motsvarande trunkerade Taylorserien ger 2.71805.

För att i allmänhet beräkna en rationell approximation försöker vi minimera avvikelsen mellan $R_N(x)$ och $f(x)$ inom ett visst intervall $[a, b]$. Om avvikelsen betecknas med $e(x) \equiv R_N(x) - f(x)$, så är dess största värde $e_{\max} = \max_{a \le x \le b} |e(x)|$. Detta leder till en *minimax* lösning, men hur finner man den?

Det finns ett teorem av Anthony Ralston, som säger att om $R_N(x)$ inte är degenererad (dvs saknar gemensamma faktorer i nämnare och täljare) så kan man välja polynomen i nämnare och täljare på ett entydigt sätt så att $e(x)$ blir minimal. För denna lösning har avvikelsen $e(x)$ $N + 2$ extremvärden inom intervallet $[a, b]$, som alla är av storleken $e_{\max}$ och har alternerande förtecken. Detta är analogt med minimax-egenskapen, som vi diskuterade i samband med Tjebysjevpolynomen i avsnitt 3.5. Felet i en n:te ordningens serieutveckling av en funktion i Tjebysjevpolynom domineras i allmänhet av den första bortlämnade termen, dvs T_{n+1}, som har $n + 2$ extremvärden med lika stora absoluta värden och alternerande förtecken. Antalet koefficienter för en rationell funktion har alltså motsvarande betydelse som antalet koefficienter för ett polynom.

Vi inser lätt, att polynomen i täljare och nämnare kan väljas så, att $R_N(x)$ stämmer exakt överens med $f(x)$ i $N+1$ punkter x_i. Om den rationella funktionen multipliceras med nämnaren fås ekvationssystemet

$$a_0 + a_1 x_i + \ldots + a_n x^n = f(x_i)(1 + b_1 x_i + \ldots + b_m x_i^m),\ i = 1, 2, \ldots, N+1$$

Detta är ett system av $N+1$ linjära ekvationer, varur koefficienterna a_i och b_i kan lösas. Om alla punkterna x_i ligger inom intervallet $[a, b]$, så finns det i allmänhet ett extremum mellan varje x_i och x_{i+1}, samt ytterligare extrema vid a och b, totalt $N+2$ stycken.

Istället för att sätta $R_N(x_i)$ och $f(x_i)$ lika i punkterna x_i, kan man istället kräva att deras avvikelse överensstämmer med något givet värde y_i genom att lösa det linjära ekvationssystemet

$$a_0 + a_1 x_i + \ldots + a_n x^n = [f(x_i) - y_i](1 + b_1 x_i + \ldots + b_m x_i^m),\ i = 1, 2, \ldots, N+1$$

Man kan t.ex. välja punkterna x_i så att de överensstämmer med extremställena, istället för nollställena. Då blir $y_i = \pm e_{\max}$, då man beaktat att förtecknen alternerar.

Antalet ekvationer blir nu $N+2$, eftersom $e_{\max}$ också är obekant. Vi kan alltså beräkna polynomens koefficienter då man känner extremställena. Detta leder till iterativa procedurer, som kallas *Remes algoritmer* [2], men vi skall inte närmare gå in på dem här.

[2] A. Ralston, Numerische Mathematik **7** (1965) 322

3.9. Bitvist polynomisk approximation

Vid polynomisk interpolation låter man vanligen samma polynom passera genom alla datapunkter. Som vi sett, leder detta till problem med polynom av hög ordning. För att undvika detta har man alltsedan 1960-talet använt sig av bitvist polynomiska funktioner. I detta fall tänker man sig att definitionsintervallet för den funktion, som skall approximeras, uppdelas på en serie underintervall medels ett antal *nodpunkter*, som inte behöver vara ekvidistanta. Mellan två nodpunkter tänker man sig funktionen approximerad med ett polynom av låg ordning (vanligen ett linjärt eller ett kubiskt polynom).

Det enklaste fallet är en bitvist linjär approximationsfunktion. Som namnet antyder, är den alltid rätlinjig mellan två nodpunkter. För en mängd av diskreta datapunkter (x_i, y_i), där y_i anger ett värde av en kontinuerlig funktion, består den alltså av en bruten linje $L(x)$, som har egenskapen $L(x_i) = y_i, \quad i = 1, \ldots, n$. Allmänt kan ett godtyckligt värde av approximationsfunktionen beräknas ur formeln

$$L(x) = y_i \frac{x - x_{i+1}}{x_i - x_{i+1}} + y_{i+1} \frac{x - x_i}{x_{i+1} - x_i}, \quad \text{om} \quad x_i \leq x \leq x_{i+1}, \quad i = 1, 2, \ldots, n-1.$$

Om man väljer nodpunkterna tätare, blir approximationen bättre, vilket är önskvärt. I praktiken har man dock inte alltid möjlighet att välja nodpunkterna som man behagar. Mellan nodpunkterna är approximationsfunktionens derivata

$$L'(x) = \frac{y_i}{x_i - x_{i+1}} + \frac{y_{i+1}}{x_{i+1} - x_i} = \frac{y_{i+1} - y_i}{x_{i+1} - x_i}, \text{ om } x_i < x < x_{i+1}, \ i = 1, 2, \ldots, n-1.$$

Som vi ser, sammanfaller derivatan med funktionens differenskvot. Approximationsfunktionens derivata är således också en god approximation för funktionens derivata.

Ett problem med den bitvist linjära approximationsfunktionen är att den har hörn. Polynom av högre ordning ger upphov till jämnare approximationsfunktioner. En vanlig approximationsfunktion av detta slag är en bitvist kubisk funktion, som är ett tredjegradens polynom mellan två närliggande nodpunkter. En approximationsfunktion av detta slag som passerar genom datapunkterna behöver dock inte vara entydig mellan nodpunkterna, utan därtill krävs att derivatorna är kontinuerliga. En approximationsfunktion som har en kontinuerlig första derivata kallas *Hermites kubiska interpolant* (efter *Charles Hermite*, lärare vid l'École Polytechnique i Paris på 1800–talet). Om ytterligare den andra derivatan är kontinuerlig, kallas approximationsfunktionen för en kubisk spline–funktion. Däremot lönar det sig inte att också kräva att den tredje derivatan är kontinuerlig, eftersom den är konstant.

En metod att definiera en kubisk interpolant är att välja $2n$ stycken basfunktioner $c_i(x)$ och $\hat{c}_i(x)$, $i = 1, 2, \ldots, n$, som alla antas vara bitvist kubiska funktioner med kontinuerliga derivator inom intervallet $[x_1, x_n]$. Om vi definierar

$$c_i(x_i) = 1, \qquad c_i(x_j) = 0, \qquad i \neq j, \qquad \text{och} \quad \hat{c}_i(x_j) = 0, \quad \forall\, i, j,$$

så är funktionen

$$C(x) = \sum_{i=1}^{n} [y_i c_i(x) + d_i \hat{c}_i(x)]$$

en hermitesk kubisk interpolant för alla värden av d_i. För derivatorna kan vi ytterligare uppställa villkoren

$$c_j'(x_i) = 0, \quad \forall i, j \qquad \text{och } \hat{c}_i'(x_i) = 1, \quad \hat{c}_i'(x_j) = 0, \quad j \neq i,$$

som leder till

$$C'(x_k) = \sum_{i=1}^{n} [y_i c_i'(x_k) + d_i \hat{c}_i'(x_k)] = d_k \hat{c}_k'(x_k) = d_k.$$

En sådan approximationsfunktion är lämplig att använda, om man förutom funktionsvärdena, även känner värdena av funktionens derivata i datapunkterna.

Om värdena av parametern d_i är obekanta, kan man uppskatta dem utgående från datapunkterna (eftersom $d_i = C'(x_i)$), tex genom att tänka sig en parabel dragen genom de tre punkterna (x_{i-1}, y_{i-1}), (x_i, y_i), och (x_{i+1}, y_{i+1}). Dess derivata i punkten x_i är då

$$\delta_i = \frac{\Delta_i h_{i-1} + \Delta_{i-1} h_i}{h_{i-1} + h_i}, \qquad \text{om } \Delta_i = \frac{y_{i+1} - y_i}{h_i},$$

där h_i är intervallängden. Som approximation för $d_2, \ldots, d_{n-1}$ kan vi då använda δ_i, och Δ_1 och Δ_n ger approximationer för d_1 resp. d_n.

3.10. Spline–interpolation

I många fall vill man ha en jämnare interpolationsfunktion än vad man kan åstadkomma med en vanlig kubisk interpolant. En kubisk interpolant som är definierad över intervallet $[x_1, x_n]$, som är uppdelat i $n-1$ underintervall, har exakt n fria parametrar[3] , vilket stämmer överens med antalet parametrar d_i. Om man kräver, att andra derivatan $C''(x)$ också skall vara kontinuerlig i intervallgränserna, får man ytterligare $n-2$ bindvillkor. Vid en spline–interpolation har man alltså ännu kvar två fria parametrar. Ordet spline är lånat från engelskan, där det är en benämning på en elastisk mall (använd av bl.a. skeppsbyggare), som kan böjas så att den passerar genom ett antal givna punkter.

Att beräkna spline–interpolanter är tekniskt mer krävande än vanliga kubiska interpolanter. Vi måste uttrycka $C''(x)$ i allmän form, och därpå lösa det lineära ekvationssystem, som följer av villkoret, att andra derivatan skall vara kontinuerlig i nodpunkterna. Ekvationssystemet har dock ingen entydig lösning, emedan vi har $n-2$ ekvationer med n obekanta (d_i). För att få en entydig lösning bör vi alltså uppställa några tilläggsvillkor. Exempel på sådana villkor är:

[3] Varje polynom $C(x)$ har fyra koefficienter (totalt $4(n-1)$ parametrar), dess värden i nodpunkterna skall stämma med funktionens värden ($2(n-1)$ villkor) samt derivatorna skall vara kontinuerliga i de inre ändpunkterna ($n-2$ villkor). Differensen blir därför n.

1) Ange derivatan av $C(x)$ i punkterna x_1 och x_n. I detta fall får vi en *fullständig kubisk spline–interpolant*.

2) Uppskatta derivatavärden ur datapunkterna (*försiktigt!*), och beräkna härur d_1 och d_n. Beräkna t.ex ett uttryck för det kubiska polynom som passerar genom de fyra första datapunkterna, och beräkna därpå derivatan i x_1 och sätt d_1 lika med dess värde. Upprepa proceduren för x_n.

3) Sätt $C''(x_1) = 0 = C''(x_n)$. På detta sätt fås en *naturlig spline–interpolant*, vilket i praktiken innebär, att spline–funktionen är rätlinjig utanför dataområdet.

4) Fordra, att $C'''(x)$ är kontinuerlig i punkterna x_2 och x_{n-1} (*icke-nodvillkoret*). Detta innebär, att de kubiska funktionerna är identiska inom första och andra delintervallet, resp. det $n-1$:a och n:te delintervallet.

Om vi använder något av dessa villkor blir koefficientmatrisen för ekvationssystemet en symmetrisk, icke–singulär tridiagonal matris, och systemets lösning kräver ingen pivotering. Då spline–funktionen är entydigt bestämd genom spline–koefficienterna d_i, kan den beräknas för alla värden av x.

Ett allmänt sätt att definiera en spline–funktion är följande: Antag, att det slutna intervallet $[a, b]$ uppdelas på en räcka punkter $x_0, x_1, x_2, \ldots x_n$ som uppfyller villkoret $a = x_0 < x_1 < \ldots < x_n = b$. En p:te gradens spline–funktion med noder i punkterna $x_i, i = 0, 1, 2, \ldots, n$ betecknar då en funktion s med följande egenskaper:

a) Inom varje delintervall $[x_i, x_{i+1}]$, $i = 0, 1, 2 \ldots, n-1$ är $s(x)$ ett p:te gradens polynom.
b) $s(x)$ och de $(p-1)$:a derivatorna är kontinuerliga inom intervallet $[a, b]$.

Av denna definition följer direkt, att om $s_1(x)$ och $s_2(x)$ är två spline–funktioner av samma grad, så gäller detta också för en godtycklig linjär kombination $\alpha s_1(x) + \beta s_2(x)$. Spline–funktionerna kommer således att generera ett linjärt rum.

Teorin för de kubiska spline–funktionerna baserar sig på följande sats, som inte här kommer att bevisas. Vi inför beteckningarna

$$x_i - x_{i-1} = h_i, \qquad \frac{y_i - y_{i-1}}{h_i} = d_i,$$

$$\frac{x - x_{i-1}}{h_i} = t \qquad \text{om} \quad x \in [x_{i-1}, x_i].$$

Varje kubisk spline–funktion med noderna x_i, $i = 0, 1, 2 \ldots, n$, som interpolerar de givna värdena y_i i nodpunkterna, kan inom intervallet $[x_{i-1}, x_i)$ beskrivas som ett polynom av formen

$$q_i(x) = ty_i + (1-t)y_{i-1} + h_i t(1-t)[(k_{i-1} - d_i)(1-t) - (k_i - d_i)t] \quad (i = 1, 2, \ldots, n),$$

där $k_0, k_1, \ldots, k_n$ satisfierar det tridiagonala ekvationssystemet

$$h_{i+1}k_{i-1} + 2(h_i + h_{i+1})k_i + h_i k_{i+1} = 3(h_i d_{i+1} + h_{i+1} d_i) \quad (i = 1, 2, \ldots, n-1).$$

Emedan ovanstående ekvationssystem har $n+1$ obekanta, men innehåller endast $n-1$ ekvationer, behövs det ytterligare två villkor för att man entydigt skall kunna bestämma spline–funktionen, som vi ovan har givit exempel på.

Då man använder en biblioteksrutin (t.ex. `spline` i MATLAB) för att beräkna en spline–funktion, är det viktigt att studera dokumentationen för att komma underfund med hur spline–koefficienterna beräknas. Genom jämförelse med den funktion, som använts för att beräkna datapunkterna, kan man få en uppfattning om felet vid approximationen. Om denna funktion har en kontinuerlig fjärde derivata, så kan man visa, att felet är proportionellt mot fjärde potensen av delintervallens längd, vilket säger något om snabbheten varmed felen minskar, då delningen görs tätare. Om den ursprungliga funktionen är tillräckligt jämn, kan man uppskatta dess derivator ur spline–funktionens derivator. Spline–funktioner kan också användas för flerdimensionell interpolation.

Spline–funktionerna är i viss mån "överreklamerade". Det finns många exempel, bl.a. ur fysiken, på att spline—approximation kan ge dåliga resultat, speciellt vid snabba förändringar i datapunkterna. Vid analys av spektra lönar det sig sålunda att använda spline–funktioner endast för att approximera bakgrunden. För att undgå detta problem har man på senare tid försökt införa en "spänningsparameter" σ (analogt med föreställningen om ett elastiskt band), och inkludera funktioner av typen $e^{\sigma x}$ och $e^{-\sigma x}$ i approximationsfunktionen. I allmänhet kan man säga, att det alltid lönar sig att rita en graf av interpolationsfunktionen, och jämföra den med datapunkterna, för att kunna avgöra om approximationen är förnuftig.

3.11. Bézier–kurvor och B-spline–funktioner

Vid datorstödd design (CAD), som numera är en ytterst viktig metod vid industriell formgivning, gäller det att interaktivt pröva olika parametrar, tills det grafiska mönstret ser "riktigt" ut. År 1962 utvecklades program för detta av fransmännen Bézier och de Casteljau vid Renault– och Citröen–verken.

Metoden bygger på Bernstein–polynomen, som vi redan tidigare berört i samband med Weierstrass' approximationsteorem. Om vi antar, att datapunkterna är givna inom det slutna intervallet $[a, b]$, så kan den i:te Bernstein–funktionen , som är ett n:te gradens polynom, uttryckas som

$$B_i^n(x) = \binom{n}{i} \frac{(b-x)^{n-i}(x-a)^i}{(b-a)^n}, \qquad i = 0, 1, \ldots, n,$$

med den vanliga beteckningen för binomialkoefficienten

$$\binom{n}{i} \equiv \frac{n!}{i!(n-i)!}.$$

Om $n = 3$, så blir de fyra kubiska Bernstein–funktionerna

$$B_0^3(x) = \frac{(b-x)^3}{(b-a)^3}, \qquad B_1^3(x) = \frac{3(b-x)^2(x-a)}{(b-a)^3}$$
$$B_2^3(x) = \frac{3(b-x)(x-a)^2}{(b-a)^3}, \qquad B_3^3(x) = \frac{(x-a)^3}{(b-a)^3}.$$

För att beräkna värden av Bernstein–funktionerna använder man rekursionslikheterna

$$\begin{aligned} B_0^0(x) &= 1, \qquad B_{-1}^n(x) = 0, \\ B_i^n(x) &= (b-x)B_i^{n-1}(x) + (x-a)B_{i-1}^{n-1}(x)(b-a), \quad 0 \le i \le n \\ B_{n+1}^n(x) &= 0 \end{aligned}$$

som är stabilare.

Eftersom alla polynomen B_i^n är av n:te graden, så gäller detta även den linjära kombinationen

$$P_n(x) = \sum_{i=0}^{n} p_i B_i^n(x).$$

Koefficienterna p_i kan beräknas genom att lösa det linjära ekvationssystem som bildas, om man antar, att $P_n(x_i)$ sammanfaller med de $n+1$ datapunkterna y_i.

Sambandet mellan koefficienterna p_i och datapunkterna y_i visar sig bli rätt så enkelt. Man kan lätt visa, att för $P_n(x)$ kommer de första och sista koefficienterna p_0 och p_n att sammanfalla med datapunkterna i intervallets ändpunkter. Också de övriga koefficienterna kan relateras till datapunkterna, t.o.m. genom att man gissar sig fram, så att man kan få en föreställning om utseendet av $P_n(x)$ redan innan man räknat ut den! I praktiken görs detta genom att man väljer ett antal kontrollpunkter i planet. Om man låter $\mathbf{p}_i$ vara en punkt i planet, kan man definiera en *vektoriell* Bézier–kurva

$$\mathbf{P}_n(x) = \sum_{i=0}^{n} \mathbf{p}_i B_i^n(x).$$

$\mathbf{P}_n$ kommer då att följa en kurva i planet. Bézier utvidgade sin metod också till tredimensionella ytor genom att generera kartesiska produkter av två kurvor.

Också om Béziers kurvor och ytor lämpar sig utmärkt för att lösa olika formgivningsproblem, behöver man rätt så komplicerade geometriska konstruktioner för att kontinuiteten skall bevaras vid gränsen mellan två kurvor eller ytor. Dessa problem kan undvikas genom att man använder spline–funktioner.

Vi har tidigare visat, att en kubisk spline–funktion kan uttryckas i formen

$$C(x) = \sum_i [y_i c_i(x) + d_i \hat{c}_i(x)],$$

där koefficienterna d_i kan beräknas genom att man löser ett linjärt ekvationssystem. Varken c_i eller $\hat{c}_i$ är dock spline–funktioner, eftersom de inte kan deriveras två gånger. Många gånger skulle det vara bättre att uttrycka approximationsfunktionen som en linjär kombination av spline–funktioner:

$$C(x) = \sum_i a_i N_i^3(x).$$

För detta ändamål används vanligen en speciell typ av kubiska spline–funktioner, som kallas B-spline–funktioner ('*B*-' kommer av det engelska ordet *B*ellshaped, eller *B*asic). En kubisk B-spline–funktion $N_i^3(x)$ för ekvidistanta noder $x_i, \ldots, x_{i+4}$ kan definieras på följande sätt:

$$N_i^3(x) = \frac{1}{6h^3} \begin{cases} (x - x_i)^3 & , x_i \leq x < x_{i+1} \\ h^3 + 3h^2(x - x_{i+1}) + 3h(x - x_{i+1})^2 - 3(x - x_{i+1})^3 & , x_{i+1} \leq x < x_{i+2} \\ h^3 + 3h^2(x_{i+3} - x) + 3h(x_{i+3} - x)^2 - 3(x_{i+3} - x)^3 & , x_{i+2} \leq x < x_{i+3} \\ (x_{i+4} - x)^3 & , x_{i+3} \leq x < x_{i+4} \end{cases}$$

För alla övriga värden av x försvinner spline–funktionen. Man kan då visa, att en godtycklig kubisk spline–funktion entydigt kan framställas med B-spline–funktioner inom intervallet $[x_0, x_n]$ på följande sätt:

$$C(x) = \sum_{i=-3}^{m+1} a_i N_i^3(x).$$

Av denna ekvation följer att B-spline-funktionens koefficienter är analoga med Bézier–kurvans koefficienter. Om man låter koefficienterna svara mot godtyckliga punkter i planet får man de såkallade Bézier–B-spline-funktionerna. En förändring av B-spline–funktionens koefficienter kommer därvid att ha samma effekt, som om man skulle förändra en Bézier–kurvas kontrollpunkter.