Utilizarea Exemplificată a Funcţiilor de Aproximare

Pantelimon George Popescu
Radu Dumitru Stochiţoiu

pentru $MIKE + 5$

Prefaţă

Aceasta lucrare isi propune sa prezinte sumar, dar bine exemplificat, unele dintre cele mai importante si utilizate concepte de analiza numerica, interpolari si aproximari. Cartea se adreseaza unui public larg care, fara prea multa matematica si rigurozitate, poate intelege conceptele propuse, dar, si mai important, cum se pot utiliza in aplicatii practice reale.

Cartea este structurata in doua parti, una in care se prezinta pe scurt cele mai generale si uzuale interpolari si aproximari, si alta in care aceste metode numerice sunt aplicate in exemple concrete din realitate. In prima parte, fara a fi prea rigurosi, sunt discutate si comentate prin exemple punctuale, metodele implicate. In partea a doua s-a considerat un grup de exemple reale pentru care s-au aplicat concret aceste metode, obtinandu-se date numerice reale, care au fost mai apoi comparate cu efectele din realitate ale exemplelor considerate. Fiecare exemplu a fost comentat detaliat si pentru fiecare s-a furnizat codul MATHEMATICA, prin care s-au prelucrat datele, evidentiind prin grafice relevante, la fiecare pas, efectul aplicarii unei metode sau a alteia.

Nu de putine ori, chiar si noi am fost surprinsi de cat de multa informatie se poate obtine analizand un singur parametru al unui sistem complex si cat de precis poate fi rezultatul cand metodele implicate sunt utilizate corect.

In consecinta, va invitam sa parcurgeti aceasta lucrare, in special partea a doua a ei, pentru a va convinge de cat de usor si precis pot fi aproximati diversi parametri ai unui sistem si ce rezultate, poate incredibile, se pot obtine.

Autorii

Cuprins

Capitolul 1

Interpolări şi Aproximări Polinomiale

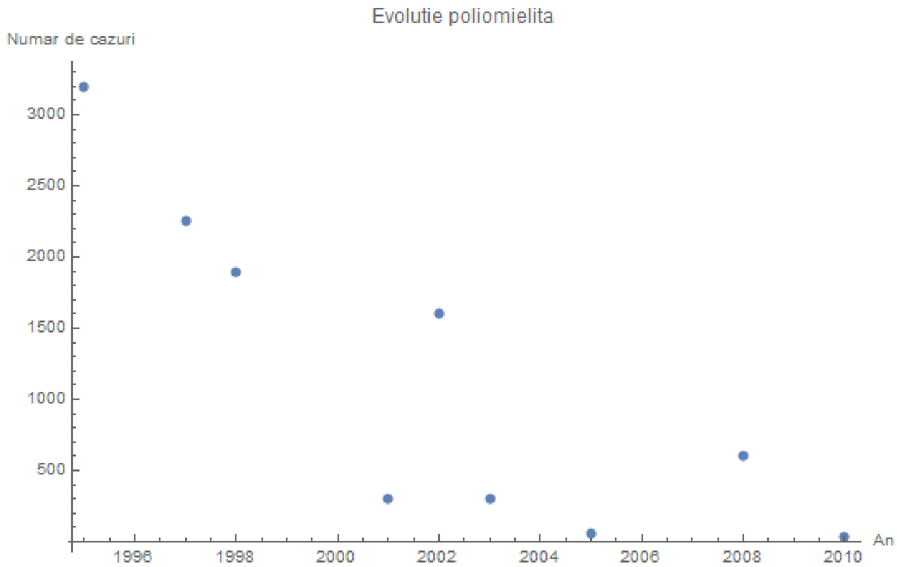
1.1 Interpolare

In domeniul stiintelor exacte si al analizei numerice, interpolarea este o metoda sau un algoritm de generare a unor puncte necunoscute initial, folosind o multime discreta de puncte cunoscute.

Vom considera urmatorul exemplu pentru a intelege practic conceptul de interpolare.

Poliomielita este o boala care si-a micsorat numarul de victime cu 99% din 1988 pana in 2013. Tabelul urmator contine cateva date referitoare la acest virus.

An	1995	1997	1998	2001	2002	2003	2005	2008	2010
Cazuri	3200	2250	1900	300	1600	300	60	600	30

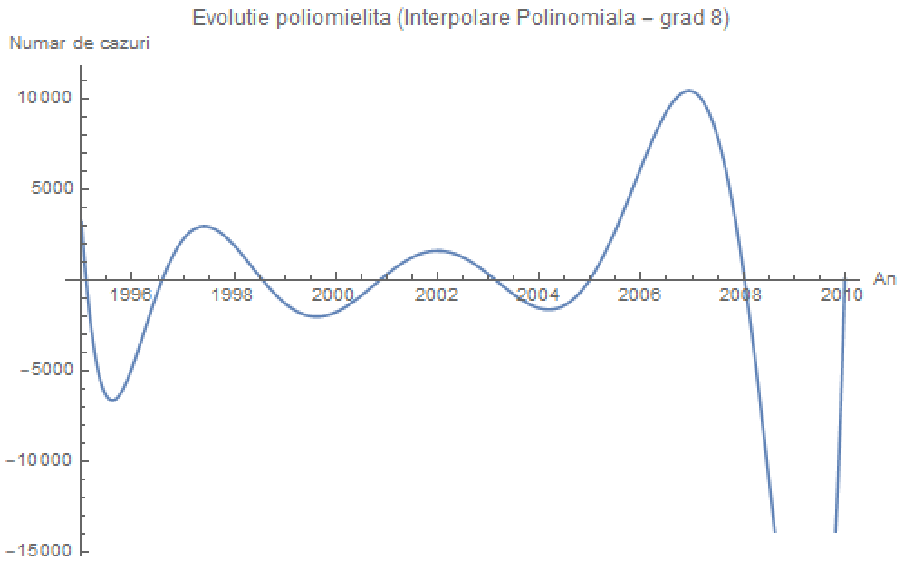


Ne-ar interesa o aproximare a numarului de persoane infectate dintr-un an pe care nu il avem in tabel din intervalul 1995-2010 (de exemplu, 2000). Acesta este rolul unei interpolari.

Interpolarea polinomiala este o interpolare care aproximeaza functia cunoscuta intr-un set de puncte cu un polinom.

Avand o multime discreta de $n+1$ puncte de forma (x_i, y_i) care apartin unei functii $f\colon [a,b] \to \mathbb{R}$, astfel incat $y_i = f(x_i)$, cautam un polinom $P_n(x) = a_n x^n + a_{n-1}x^{n-1} + ... + a_2x^2 + a_1x + a_0$ care sa treaca prin toate punctele.

Ne-am dori o interpolare polinomiala din motive computationale, operatiile de integrare si derivare la nivelul unui calculator fiind mult mai putin costisitoare decat la alte tipuri de functii (nepolinomiale).



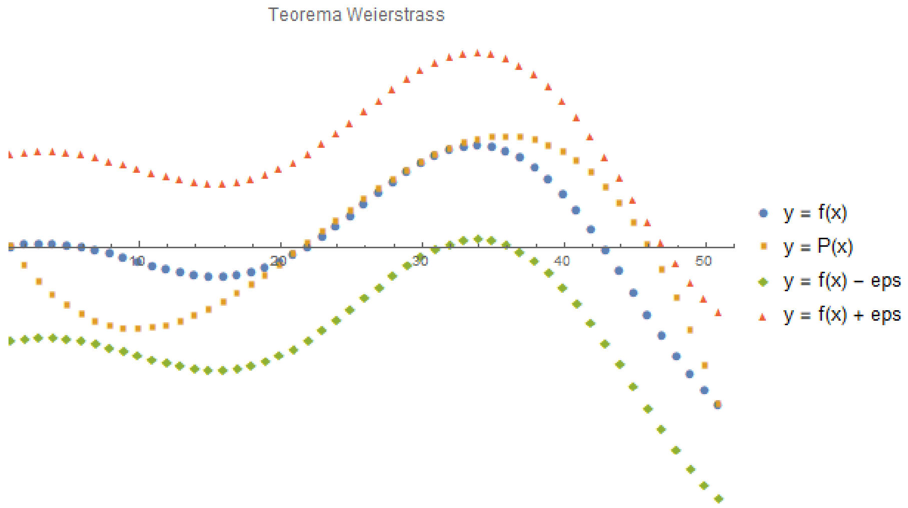
O interpolare care nu ia in calcul erorile de aproximare poate fi mai indepartata de realitate decat ne-am astepta, insa tinem cont de avantajele enumerate anterior in sensul eficientei computationale.

Observam pentru exemplul nostru ca intre anii 2008 si 2010 polinomul de interpolare ia valori negative si foarte mici, lucru care pentru exemplul practic pe care l-am ales nu are sens, insa din punct de vedere matematic este perfect valid. Aflam astfel ca in acest domeniu al aproximarilor de functii este necesar nu numai un fundament matematic, dar si sa avem o interpretarea corecta a acestuia.

Rezultatul care ne permite sa aproximam printr-un polinom este dat de Teorema lui Weierstrass[1].

Teorema *(Teorema lui Weierstrass) Fie* $f\colon [a,b] \to \mathbb{R}$ *o functie.* $\forall \epsilon > 0$, $\exists P(x)$, *astfel incat* $|f(x) - P(x)| < \epsilon, \forall x \in [a,b]$.

[1]Karl Theodor Wilhelm Weierstrass, mat. german, 1815-1897



Interpolare Lagrange

Daca $x_0, x_1, ..., x_n$ sunt $n+1$ puncte distincte in care functia f este cunoscuta, atunci exista un polinom unic de grad cel mult n, $P(x)$, care satisface relatia $P(x_i) = f(x_i)$ si are forma:

$$P_n(x) = \sum_{k=0}^{n} f(x_k) \cdot \prod_{i=0, i\neq k}^{n} \frac{x - x_i}{x_k - x_i}.$$

Aceasta interpolare poarta numele de interpolare Lagrange[2].

Polinoamele

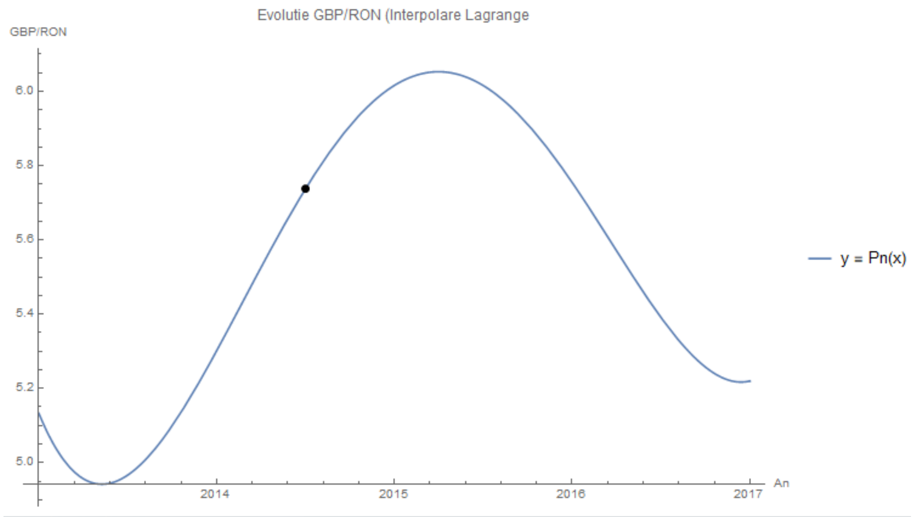
$$l_k(x) = \prod_{i=0, i\neq k}^{n} \frac{x - x_i}{x_k - x_i}$$

se numesc multiplicatori Lagrange. Coeficientii polinomului de interpolare Lagrange pot fi calculati prin adunarea a $n+1$ polinoame de gradul n, obtinute prin dezvoltarea produselor ce apar in $P_n(x)$, definit anterior.

[2]Joseph-Louis Lagrange, mat. italian, 1736-1813

Continuam cu un alt exemplu, si anume: evolutia raportului GBP/RON in ultimii cinci ani (pe 1 ianuarie) arata ca in tabelul de mai jos.

An	2013	2014	2015	2016	2017
GBP/RON	5.133	5.301	6.016	5.755	5.221



Ne propunem sa aflam raportul la mijlocul anului 2014, cand valoarea lirei sterline era in crestere fata de valoarea leului romanesc. Aproximarea cu ajutorul polinomului Lagrange este 5.7375, valoarea reala fiind 5.6577. Bineinteles, daca am fi aproximat eroarea mai bine sau daca am fi folosit mai multe puncte in suportul interpolarii (punctele in care functia este cunoscuta initial), valoarea aflata prin interpolare ar fi fost, posibil, mult mai apropiata de cea reala.

Observam ca folosind doar cinci puncte in suportul interpolarii obtinem un rezultat satisfacator. Acest fapt ne arata cat de puternice sunt metodele de interpolare sau aproximare alese.

Continuam prin a prezenta un rezultat care evidentiaza eroarea interpolarii Lagrange.

Teorema *Daca alegem un inte*[illegible] $[a, b]$ *si o functie* $f \in \mathbb{C}^{n+1}$ *in care cunoastem* $n+1$ *puncte distincte,* $x_0, x_1, ..., x_n$*, atunci* $\forall x \in [a, b], \exists \xi(x) \in$

$[x_0, x_n]$ *astfel incat:*

$$f(x) = P_n(x) + \frac{f^{n+1}(\xi(x))}{(n+1)!}(x - x_0)(x - x_1)(x - x_2)...(x - x_n),$$

unde $P_n(x)$ *este polinomul Lagrange asociat suportului de interpolare cunoscut.*

Aceasta teorema este folositoare deoarece permite aflarea unor margini ale functiei, asadar se pot face aproximari mult mai bune in momentul derivarii sau integrarii, domeniu in care polinoamele Lagrange sunt folosite deosebit de mult.

Ca si exemplu, sa presupunem ca un bolid se misca cu o acceleratie $x'' = \ln x^3$ la plecare, insa intre secundele 2 si 3, calculatorul de bord primeste doar datele corespunzatoare momentelor de timp $x_0 = 2, x_1 = 2.5, x_3 = 3$. Pentru a putea face masuratori corecte, dorim sa aproximam valorile acceleratiei in mai multe puncte dintre 2 si 3, insa am dori, de asemenea, sa stim si eroarea la care e expusa metoda de interpolare, coordonarea si adunarea datelor unui bolid fiind o operatiune foarte sensibila si importanta pentru siguranta si confortul soferului si pasagerilor.

Notand acceleratia cu $f(x)$, obtinem urmatoarele relatii:

$$f(x) = \ln x^3$$

$$f'(x) = \frac{3}{x}$$

$$f''(x) = \frac{3}{-x^2}$$

$$f'''(x) = \frac{6}{x^3}$$

Dupa cum am mentionat in teorema precedenta, putem scrie eroarea interpolarii polinomiale Lagrange ca fiind:

$$\frac{f'''(\xi(x))}{3!}(x - x_0)(x - x_1)(x - x_2) = \frac{1}{\xi^3(x)}(x - 2)(x - 2.5)(x - 3)$$

Notand cu $g(x) = (x-2)(x-2.5)(x-3)$, pentru a afla punctele critice ii vom egala derivata cu 0, rezultand doua puncte critice:

$$\begin{Bmatrix} g_0 = 2.21132 \\ g_1 = 2.78868 \end{Bmatrix}$$

$$|(x-x_0)(x-x_1)(x-x_2)| \leq 0.048113$$

si stiind ca $\xi(x) \in [2,3]$, vom alege $\xi(x) = 2$ pentru a maximiza eroarea, rezultand:

$$\frac{f'''(\xi(x))}{3!}(x-x_0)(x-x_1)(x-x_2) \leq 0.0060141$$

Observam ca avem o eroare de ordinul 10^{-3} folosind doar trei puncte in care cunoastem functia, dovedind din nou importanta datelor utile in suportul interpolarii.

Metoda Neville

Din pacate, in general nu avem datele necesare pentru a calcula eroarea interpolarii Lagrange, o derivata de un ordin foarte mare fiind dificil de obtinut. De asemenea, avand efectuat calcule cu polinoame de ordin foarte mare, va creste complexitatea computationala a interpolarii Lagrange. Tinand cont de aceste considerente, s-a decis sa se foloseasca recursiv polinoame de grade mai mici pentru a forma din ce in ce mai putine polinoame de grade mai mari, rezultand intr-un final o interpolare ce foloseste suportul de puncte existent.

Plecand de la fiecare punct cunoscut, aproximat ca un polinom de grad 0, se imbina cate doua polinoame adiacente pe rand intr-un polinom de grad superior, la fiecare iteratie.

Relatia rezultata este una piramidala, precum:

$$\begin{array}{lllll}
P_{0,0}(x) = f(x_0) & & & & \\
P_{1,1}(x) = f(x_1) & P_{0,1}(x) & & & \\
P_{2,2}(x) = f(x_2) & P_{1,2}(x) & P_{0,2}(x) & & \\
\quad ... & \quad ... & \quad ... & ... & \\
P_{n,n}(x) = f(x_n) & P_{n-1,n}(x) & P_{n-2,n}(x) & ... & P_{0,n}(x)
\end{array}$$

Teorema *Daca notam* $\sigma = \{i_1, i_2, ..., i_p\}$ *si* P_σ *polinomul de interpolare care trece prin punctele* $\{(x_{i_1}, y_{i_1}), (x_{i_2}, y_{i_2}), ..., (x_{i_p}, y_{i_p})\}$, *atunci polinomul de interpolare pentru ansamblul extins* $\sigma + j + k = \sigma \cup \{j, k\}$ *se defineste astfel:*

$$P_{\sigma+j+k}(x) = \frac{(x - x_j)P_{\sigma+k}(x) - (x - x_k)P_{\sigma+j}(x)}{x_k - x_j}.$$

Se observa ca $P_{\sigma+j+k}(x_i) = y_i, \forall i \in \sigma + j + k$.

Particularizand, daca $x_0, x_1, ..., x_n$ sunt $n+1$ puncte distincte in care functia f este cunoscuta, atunci exista un polinom unic de grad cel mult n, $P(x)$, care satisface relatia $P(x_i) = f(x_i)$ si care poate fi calculat recursiv plecand de la cazul de baza $P_{i,i} = f(x_i)$, astfel:

$$P_{i,j}(x) = \frac{(x_j - x)P_{i,j-1}(x) + (x - x_i)P_{i+1,j}(x)}{x_j - x_i}.$$

Aceasta metoda poarta numele de interpolare Neville[3]. Valoarea polinomului de interpolare Neville se gaseste in $P_{0,n}(x)$.

Pentru exemplul de mai sus, daca dorim sa aflam valoarea polinomului Neville intr-un punct dintre $x_0 = 2$ si $x_2 = 3$, de exemplu 2.2, metoda Neville se va efectua astfel:

$$\begin{aligned}
P_{0,0}(2.1) &= f(x_0) = \ln 2^3 \approx 2.079 \\
P_{1,1}(2.1) &= f(x_1) = \ln 2.5^3 \approx 2.748 \\
P_{2,2}(2.1) &= f(x_2) = \ln 3^3 \approx 3.295 \\
P_{0,1}(2.1) &= \frac{(x_1 - 2.1)P_{0,0}(2.1) + (2.1 - x_0)P_{1,1}(2.1)}{x_1 - x_0} \approx 2.213 \\
P_{1,2}(2.1) &= \frac{(x_2 - 2.1)P_{1,1}(2.1) + (2.1 - x_1)P_{2,2}(2.1)}{x_2 - x_1} \approx 2.310 \\
P_{0,2}(2.1) &= \frac{(x_2 - 2.1)P_{0,1}(2.1) + (2.1 - x_0)P_{1,2}(2.1)}{x_2 - x_0} \approx 2.2227
\end{aligned}$$

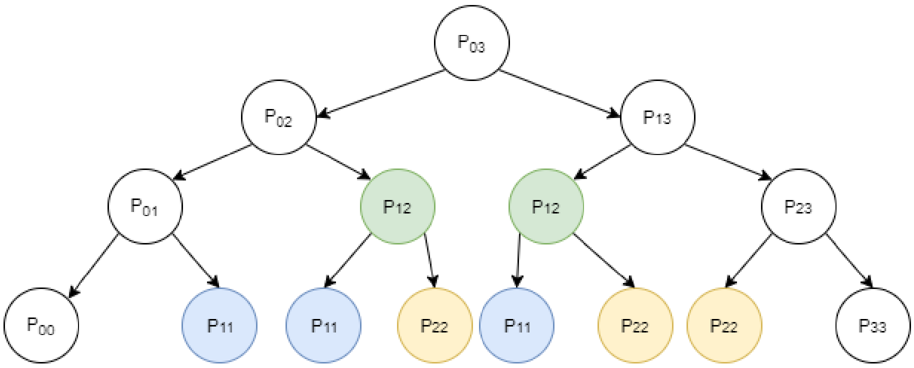
[3]Eric Harold Neville, mat. englez, 1889-1961

Valoarea reala a functiei in punctul $x = 2.1$ este $\ln 2.1^3 \approx 2.2258$. Tinand cont ca am urmarit doar trei zecimale la fiecare pas si ca eroarea nu iese din marginile calculate anterior (± 0.0060141), putem spune ca rezultatul este unul promitator.

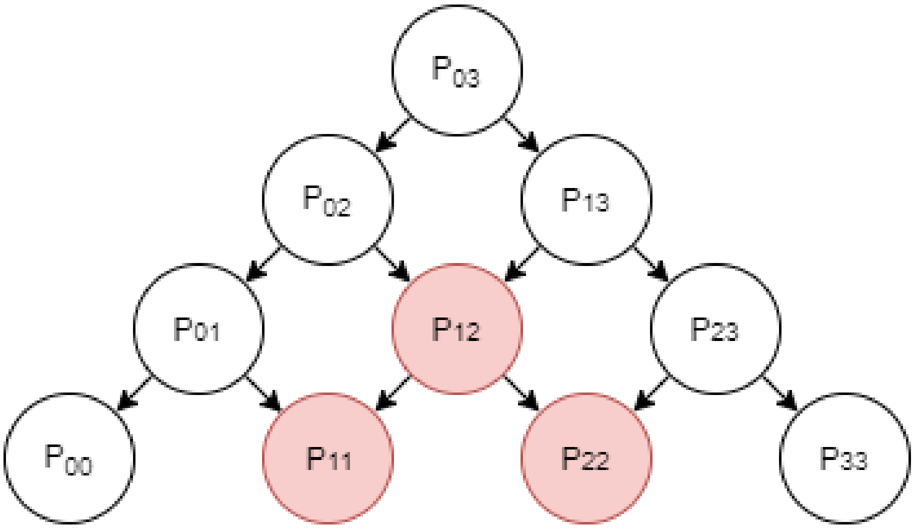
Desi se poate calcula foarte usor in modul naiv (din punct de vedere al implementarii), arborele de recursivitate se poate reduce prin memoizare (un tip de programare dinamica care implica pastrarea rezultatelor unor calcule costisitoare pentru refolosirea si evitarea recalcularii ulterioare a acestora).

De retinut ca in practica este indicat sa se plece de la o abordare naiva pe un exemplu mic si controlabil. Trecerea la o abordare optima se face dupa verificarea logicii si a corectitudinii formulei de recursivitate.

Abordarea naiva



Programare dinamica (memoizare/caching)



Se poate observa ca prin simpla memoizare a valorilor polinoamelor intermediare, doar pentru patru puncte in suportul interpolarii se reduce numarul de apelari ale functiei cu cinci. Acest rezultat depinde exponential de numarul de puncte in care este cunoscuta functia initial.

Interpolări Newton[4]

Diferenţe Divizate

O alta metoda de interpolare este Metoda Diferentelor Divizate. Aceasta presupune scrierea polinomului de aproximare ca o suma de coeficienti inmultiti pe rand cu cate o noua diferenta.

[4]Isaac Newton, mat. si fizician englez, 1642-1727

Cazul liniar

Sa presupunem ca am cunoaste valoarea unei functii doar in doua puncte: $(x_0, y_0), (x_1, y_1)$. Astfel, am putea scrie o functie $f_1(x)$ care trece liniar prin ele astfel: $f_1(x) = b_0 + b_1(x - x_0)$.

Observam ca daca verificam valoarea in punctul x_0, aceasta este $f_1(x_0) = b_0$ care se va nota mai departe cu $f_1(x_0) = F_0[x_0]$ si se va numi "diferenta divizata de ordinul 0". Verificand si valoarea in punctul x_1, aflam $b_1 = \frac{f_1(x_1)-f_1(x_0)}{x_1-x_0} = \frac{F_0[x_1]-F_0[x_0]}{x_1-x_0} = F_1[x_1, x_0]$, numindu-se "diferenta divizata de ordinul 1".

Cazul patratic

Presupunem ca de data aceasta cunoastem trei puncte: $(x_0, y_0), (x_1, y_1)$, (x_2, y_2). Astfel, putem scrie o functie patratica $f_2(x)$ care trece prin cele trei puncte: $f_2(x) = b_0 + b_1(x - x_0) + b_2(x - x_0)(x - x_1)$.

Coeficientii se vor calcula la fel, iar primii doi, b_0 si b_1 vor fi tot $F_0[x_0]$, respectiv $F_1[x_1, x_0]$. In continuare, daca verificam si valoarea din punctul x_2, aflam $b_2 = \frac{\frac{f_2(x_2)-f_2(x_1)}{x_2-x_1} - \frac{f_2(x_1)-f_2(x_0)}{x_1-x_0}}{x_2-x_0}$. Astfel, $b_2 = \frac{F_1[x_2,x_1]-F_1[x_1,x_0]}{x_2-x_0} = F_2[x_2, x_1, x_0]$, numindu-se "diferenta divizata de ordinul 2".

Cazul general

Se observa ca pentru $n + 1$ puncte cunoscute, se poate continua pe acelasi principiu si se pot calcula toti cei $n + 1$ coeficienti, formula finala demonstrandu-se prin inductie matematica.

Daca $x_0, x_1, ..., x_n$ sunt $n + 1$ puncte distincte in care functia f este cunoscuta, atunci exista un polinom unic de grad n, $P_n(x) = b_0 + b_1(x - x_0) + b_2(x - x_0)(x - x_1) + ... + (x - x_0)(x - x_1)...(x - x_n)$, care satisface relatia $P_n(x_i) = f(x_i)$ si care are coeficienti ce pot fi calculati recursiv astfel:

- $b_0 = F_0[x_0] = f(x_0)$
- $b_1 = F_1[x_1, x_0] = \frac{F_0[x_1]-F_0[x_0]}{x_1-x_0}$

- ...
- $b_p = F_p[x_p, ..., x_0] = \frac{F_{p-1}[x_{p-1},...,x_0] - F_{p-1}[x_p,...,x_1]}{x_p - x_0}$
- $b_p = F_p[x_p, ..., x_0] = \sum\limits_{k=0}^{p} \frac{f(x_k)}{\pi'(x_k)} = \sum\limits_{k=0}^{p} \frac{f(x_k)}{\prod\limits_{i=0}^{p}(x_k - x_i)}$

Diferenţe Finite

Diferente progresive (inainte)
$\Delta f(x_i) = f(x_i + h) - f(x_i) = f(x_{i+1}) - f(x_i)$
$\Delta^k f(x_i) = \Delta^{k-1} f(x_{i+1}) - \Delta^{k-1} f(x_i)$

Diferente regresive (inapoi)
$\nabla f(x_i) = f(x_i) - f(x_i - h) = f(x_i) - f(x_{i-1})$
$\nabla^k f(x_i) = \nabla^{k-1} f(x_i) - \nabla^{k-1} f(x_{i-1})$

Diferente centrate
$\delta f(x_i) = f(x_i + \frac{h}{2}) - f(x_i - \frac{h}{2}) = f(x_{i+\frac{1}{2}}) - f(x_{i-\frac{1}{2}})$
$\delta^k f(x_i) = \delta^{k-1} f(x_{i+\frac{1}{2}}) - \delta^{k-1} f(x_{i-\frac{1}{2}})$

Trecerea de la diferente finite la diferente divizate se face astfel:

$$\Delta^n f(x_i) = \nabla^n f(x_{i+n}) = \delta^n f(x_{i+\frac{n}{2}}) = n! h^n F_n[x_{i+n}, ..., x_i]$$

Daca $x_0, x_1, ..., x_n$ sunt $n + 1$ puncte distincte si echidistante ($x_{i+1} = x_i + h$, i = 0..n-1) in care functia f este cunoscuta si dorim sa aflam valoarea acesteia intr-un punct $x \neq x_i$, vom considera $x \in (x_0, x_1)$ sau $x \in (x_{n-1}, x_n)$, fara a restrange generalitatea, deoarece daca x se afla undeva la mijloc, atunci nu mai consideram primele sau ultimele valori unde functia f este cunoscuta si, astfel, reducem problema la unul din cele doua cazuri.

Caz 1 ($x \in (x_0, x_1)$)
$x = x_0 + uh, 0 < u < 1$ Aplicand *Identitatea lui Newton* in x cu diferente

divizate exprimate prin diferente progresive, adica

$$F_k[x_k, ..., x_0] = \frac{\Delta^k f(x_0)}{k!h^k},$$

obtinem:

$$P_1(x) = P_1(x_0 + uh) = p_1(u) = f(x_0) + \frac{x - x_0}{1!h}\Delta f(x_0)$$
$$+\frac{(x - x_0)(x - x_1)}{2!h^2}\Delta^2 f(x_0) + ...+$$
$$\frac{(x - x_0)(x - x_1)...(x - x_{n-1})}{n!h^n}\Delta^n f(x_0)$$

.

Deoarece

$$x - x_k = x - x_0 - (x_k - x_0) = uh - kh = (u - k)h,$$

rezulta:

$$p_1(u) = f_0 + \frac{u}{1!}\Delta f_0 + \frac{u(u-1)}{2!}\Delta^2 f_0 + ...+$$
$$\frac{u(u-1)...(u-n+1)}{n!}\Delta^n f_0,$$

de unde, considerand conceptul generalizat de combinare vom avea formula de interpolare *Newton 1*:

$$p_1(u) = f_0 + \binom{u}{1}\Delta f_0 + \binom{u}{2}\Delta^2 f_0 + ... + \binom{u}{n}\Delta^n f_0$$

Caz 2 ($x \in (x_{n-1}, x_n)$)
Acesta se imparte in doua subcazuri:

- $x = x_n - uh, 0 < u < 1$
  Aplicand *Identitatea lui Newton* in x cu diferente divizate exprimate prin diferente regresive, vom obtine formula de interpolare *Newton 2*:

$$p_2(u) = f_0 - \binom{u}{1}\nabla f_n + \binom{u}{2}\nabla^2 f_n + ... + (-1)^n\binom{u}{n}\nabla^n f_n$$

- $x = x_n + uh, -1 < u < 0$
  Aplicand *Identitatea lui Newton* in x cu diferente divizate exprimate prin diferente regresive, vom obtine formula de interpolare *Newton 3*:

$$p_3(u) = f_n + \binom{u}{1}\nabla f_n + \binom{u+1}{2}\nabla^2 f_n + ... + (-1)^n \binom{u+n-1}{n}\nabla^n f_n$$

Spline

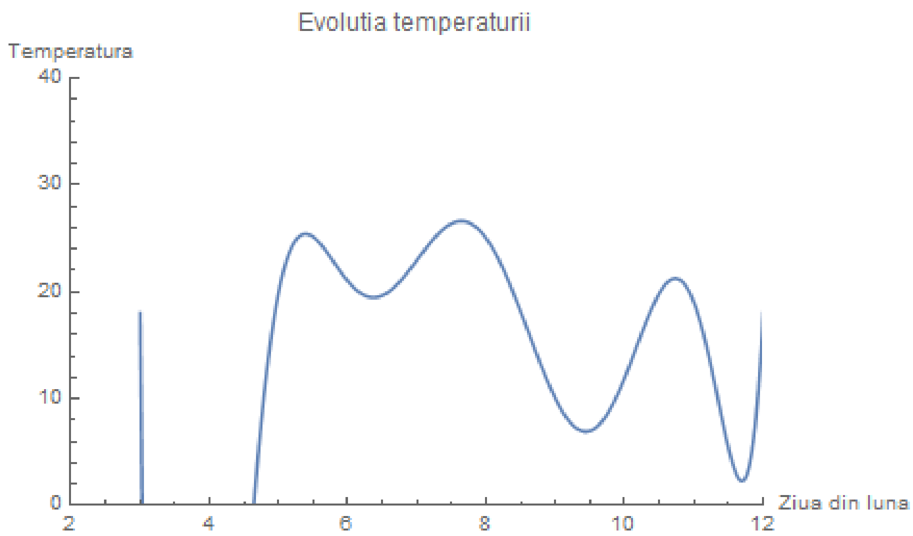
Stim ca pentru a aproxima cat [illegible] bine o functie sau un grafic, avem nevoie de cat mai multe puncte in suportul interpolarii, insa metodele precum Lagrange sau Neville creeaza cate un polinom de grad proportional cu cardinalul suportului interpolarii, deci sunt predispuse la erori foarte mari. Astfel, spline-urile propun ca in loc sa construim un polinom de grad foarte mare, sa construim cate un polinom de grad mic pentru fiecare subinterval format din cate doua puncte consecutive care se va ocupa de orice aproximare dinauntrul acelui subinterval.

Pentru a intelege ce inseamna interpolarea cu functii spline, vom incepe cu un exemplu.

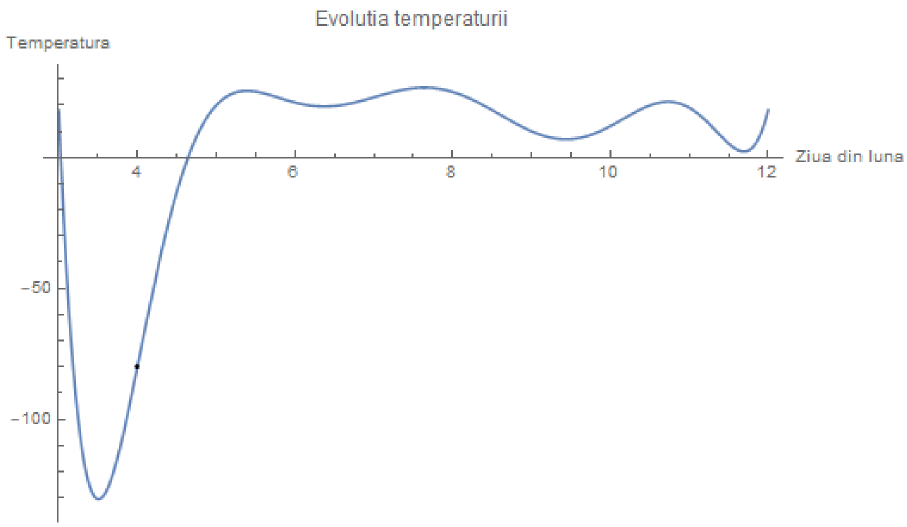
Presupunem ca avem urmatoarele date despre vremea din ultima luna:

Ziua din luna	3	5	6	7	8	9	10	11	12
Temperatura (°C)	18	20	21	23	25	10	12	19	18

Dupa cum se poate observa, lipseste temperatura din ziua a patra, asa ca ne propunem sa o aproximam. Valorea reala din acea zi este de 18°C. Vom incepe cu metoda Lagrange pe care o cunoastem deja. Graficul interpolarii Lagrange arata ca in figura de mai jos.



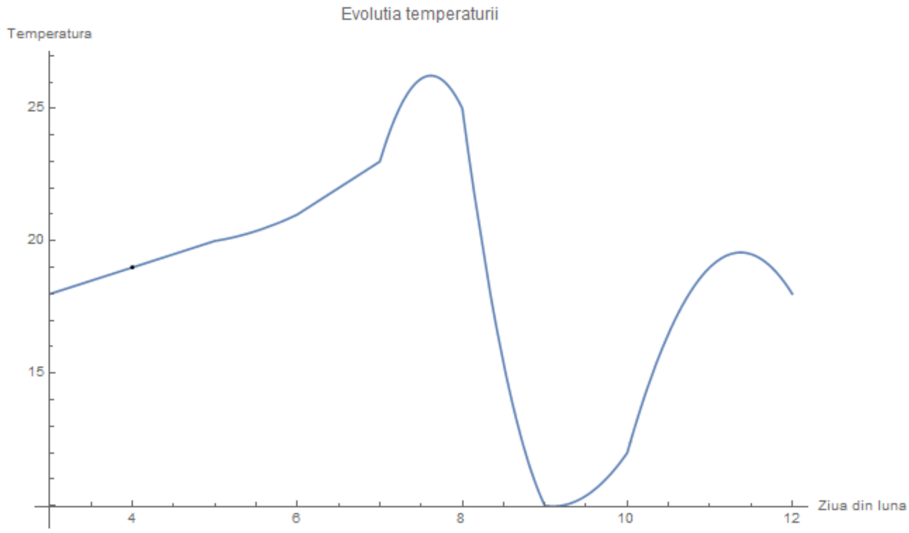
Pentru a patra zi, valoarea este negativa, iar ca sa ne putem da seama unde se afla, vom afisa intreg graficul.



Valoarea aproximata de interpolarea Lagrange este -79.6 si este mult prea departe de valoarea reala pentru a ne baza in continuare doar pe aceasta metoda de interpolare.

Vom incerca sa interpolam spline cu functii de imbinare patratice. Asta va impune ca intre fiecare doua puncte consecutive din suportul interpolarii sa existe o functie patratica, un spline, care se va ocupa de orice punct dorim sa aproximam in intervalul creat de cele doua.

Graficul arata in felul urmator.



Valoarea aproximata prin interpolare spline este 19, mult mai apropiata de cea reala, 18, decat cea obtinuta prin interpolarea Lagrange. Acest lucru ne dovedeste ca uneori, daca nu e nevoie de un polinom care sa treaca prin toate punctele si care sa depinda de ele (in cazul temperaturii, valorile de peste 10 zile depind mult mai putin de cele de acum decat de cele din vecinatatea zilei respective), putem folosi interpolarea spline si, astfel, vom avea rezultate mai apropiate de cele reale.

Un spline este o curba parametrica definita prin punctele sale de control. Se defineste printr-o functie $S : [a, b] \to \mathbb{R}$ definita local pe mai multe intervale prin functiile $S_i[x_i, x_{i+1}] \to \mathbb{R}$ cu $a = x_0 < x_1 < ... <$

$x_n = b$, unde:

$$\left\{ \begin{array}{l} S(x) = S_0(x), a = x_0 \le x < x_1 \\ S(x) = S_1(x), x_1 \le x < x_2 \\ ... \\ S(x) = S_{n-1}(x), x_{n-1} \le x < x_n \end{array} \right\}$$

De obicei:

- functiile $S_i(x)$ sunt polinoame de grad 3
- punctele x_i sunt echidistante $\Rightarrow$ *spline uniform*
- functiile spline pot fi functii de interpolare (trec prin toate punctele de control) sau de aproximare (trec doar prin unele)

Functiile pe subintervale se vor afla din *conditii de interpolare* si *conditii de racordare*. Astfel, pentru a afla aproximarea functiei intr-un punct x, prima data identificam intervalul $[x_i, x_{i+1}]$ in care se afla si apoi folosim functia specifica acelui interval, $S_i(x)$.

Tipuri de spline

Spline de clasa $\mathbb{C}^0$ *(polinoame liniare)*

Cunoastem $x_0 < x_1 < ... < x_n$ si $f(x_0), f(x_1), ..., f(x_n)$. Functiile de interpolare locale pe fiecare subinterval $[x_0, x_1), [x_1, x_2), ..., [x_{n-1}, x_n]$ vor fi liniare si vor arata astfel: $S_i(x) = a_i x + b_i, i = 0..n-1$.

Conditii de interpolare:

$S_i(x_i) = f(x_i), i = 0..n-1$
$S_{n-1}(x_n) = f(x_n)$

[4]Adobe si PostScript folosesc spline-uri ce au continuitate de clasa $\mathbb{C}^1$

Conditii de racordare:

$S_i(x_{i+1}) = S_{i+1}(x_{i+1}), i = 0..n-2$

Din aceste conditii se afla valorile coeficientilor:

$$a_i = \frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i}, i = 0..n-1$$

$$b_i = \frac{x_{i+1}f(x_i) - x_i f(x_{i+1})}{x_{i+1} - x_i}, i = 0..n-1$$

Spline de clasa $\mathbb{C}^1$ *(polinoame de grad 3 - Hermite)*

Cunoastem $x_0 < x_1 < ... < x_n$ si $f(x_0), f(x_1), ..., f(x_n)$. Functiile de interpolare locale pe fiecare subinterval $[x_0, x_1), [x_1, x_2), ..., [x_{n-1}, x_n]$ vor fi cubice si vor arata astfel: $S_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3, i = 0..n-1$.

De data aceasta, functiile se pot nota parametric astfel: $S_i(x) = a_i + b_i h_i t + c_i h_i^2 t^2 + d_i h_i^3 t^3, t \in [0,1]$, in care s-a efectuat schimbarea de variabila $t = \frac{x - x_i}{h_i}$, iar $h_i = x_{i+1} - x_i$.

Pentru micsorarea volumului de calcul, in practica, se folosesc polinoamele Bernstein[5]. Astfel, polinoamele se rescriu:

$$S_i(t) = a'_i(1-t)^3 + 3b'_i t(1-t)^2 + 3c'_i t^2(1-t) + d'_i t^3$$

Conditii de interpolare (Hermite):

$S_i(x_i) = f(x_i), i = 0..n-1$ $S'_i(x_i) = f'(x_i), i = 0..n-1$
$S_{n-1}(x_n) = f(x_n)$ $S'_{n-1}(x_n) = f'(x_n)$

Observam ca avem nevoie de functia f sa fie derivabila, de aceea metoda se numeste ”spline de clasa $\mathbb{C}^1$”.

[5] Sergei Natanovich Bernstein, mat. rus, 1880-1968

Conditii de racordare:

$S_i(x_{i+1}) = S_{i+1}(x_{i+1}), i = 0..n-2$ $S_i'(x_{i+1}) = S_{i+1}'(x_{i+1}), i = 0..n-2$

Din aceste conditii se afla valorile coeficientilor:

$$a_i' = f(x_i)$$

$$b_i' = f(x_i) + \frac{h_i}{3} f'(x_i)$$

$$c_i' = f(x_{i+1}) - \frac{h_i}{3} f'(x_{i+1})$$

$$d_i' = f(x_{i+1})$$

Spline de clasa $\mathbb{C}^2$ *(polinoame de grad 3)*

Cunoastem $x_0 < x_1 < ... < x_n$ si $f(x_0), f(x_1), ..., f(x_n)$. Functiile de interpolare locale pe fiecare subinterval $[x_0, x_1), [x_1, x_2), ..., [x_{n-1}, x_n]$ vor fi cubice si vor arata la fel ca la spline de clasa $\mathbb{C}^1$.

Conditii de interpolare (Lagrange):

$S_i(x_i) = f(x_i), i = 0..n-1$
$\Rightarrow a_i = f(x_i), i = 0..n-1$
$S_{n-1}(x_n) = f(x_n)$
$\Rightarrow a_{n-1} + b_{n-1}h_{n-1} + c_{n+1}h_{n-1}^2 + d_{n-1}h_{n-1}^3 = f(x_n) \equiv a_n$

Conditii de racordare:

$S_i(x_{i+1}) = S_{i+1}(x_{i+1}), i = 0..n-2$
$\Rightarrow a_{i+1} = a_i + b_i h_i + c_i h_i^2 + d_i h_i^3, i = 0..n-2$
$S_i'(x_{i+1}) = S_{i+1}'(x_{i+1}), i = 0..n-2$
$\Rightarrow b_{i+1} = b_i + 2c_i h_i + 3d_i h_i^2$

$$S''_i(x_{i+1}) = S''_{i+1}(x_{i+1}), i = 0..n-2$$
$$\Rightarrow c_{i+1} = c_i + 3d_ih_i$$

Tip de spline:

Natural	$S''_0(x_0) = 0$	$S''_{n-1}(x_n) = 0$
Tensionat	$S''_0(x_0) = f'(x_0)$	$S''_{n-1}(x_n) = f'(x_n)$

Din aceste conditii se afla valorile coeficientilor:

$$a'_i = f(x_i)$$

$$b'_i = \frac{a_i - a_{i-1}}{h_{i-1}} + \frac{c_{i-1} + 2c_i}{3}h_{i-1}$$

$$d'_i = \frac{c_{i+1} - c_i}{3h_i}$$

$$h_{i-1}c_{i-1} + 2(h_{i-1} + h_{i+1})c_i + h_ic_{i+1} = \frac{3(a_{i+1} - a_i)}{h_i} - \frac{3(a_i - a_{i-1})}{h_{i-1}}$$

In ambele cazuri obtinem un sistem de forma $Ax = b$.

Natural

$$A = \begin{vmatrix} 1 & 0 & 0 & 0 & \dots & 0 & 0 & 0 \\ h_0 & 2(h_0 + h_1) & h_1 & 0 & \dots & 0 & 0 & 0 \\ 0 & h_1 & 2(h_1 + h_2) & h_2 & \dots & 0 & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & 0 & \dots & h_{n-2} & 2(h_{n-2} + h_{n-1}) & h_{n-1} \\ 0 & 0 & 0 & 0 & \dots & 0 & 0 & 1 \end{vmatrix}$$

$$x = \begin{vmatrix} c_0 \\ c_1 \\ \dots \\ c_n \end{vmatrix}$$

$$b = \begin{vmatrix} 0 \\ \frac{3}{h_1}(a_2 - a_1) - \frac{3}{h_0}(a_1 - a_0) \\ \dots \\ \frac{3}{h_{n-1}}(a_n - a_{n-1}) - \frac{3}{h_{n-2}}(a_{n-1} - a_{n-2}) \\ 0 \end{vmatrix}$$

Tensionat

$$A = \begin{vmatrix} 2h_0 & h_0 & 0 & 0 & \dots & 0 & 0 & 0 \\ h_0 & 2(h_0+h_1) & h_1 & 0 & \dots & 0 & 0 & 0 \\ 0 & h_1 & 2(h_1+h_2) & h_2 & \dots & 0 & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & 0 & \dots & h_{n-2} & 2(h_{n-2}+h_{n-1}) & h_{n-1} \\ 0 & 0 & 0 & 0 & \dots & 0 & h_{n-1} & 2h_{n-1} \end{vmatrix}$$

$$x = \begin{vmatrix} c_0 \\ c_1 \\ \dots \\ c_n \end{vmatrix}$$

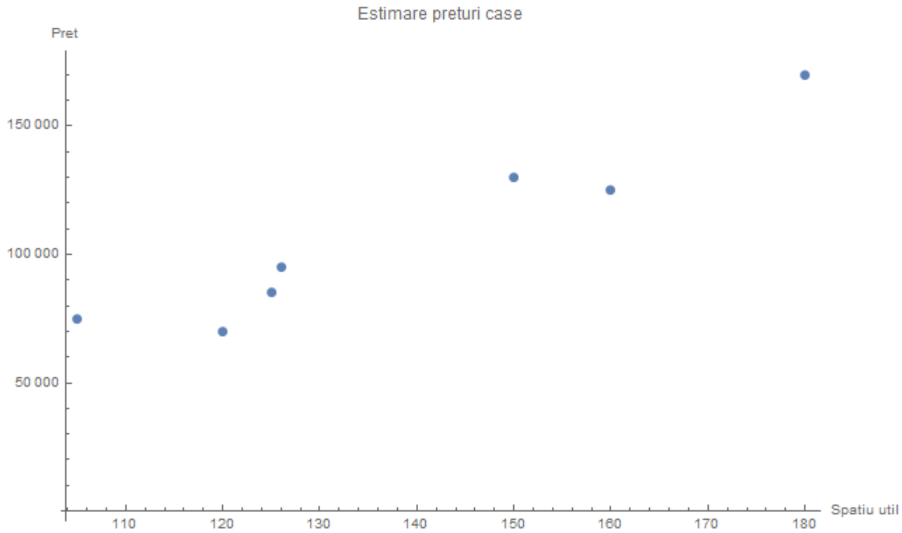
$$b = \begin{vmatrix} \frac{3}{h_0}(a_1 - a_0)) - 3f'(a) \\ \frac{3}{h_1}(a_2 - a_1) - \frac{3}{h_0}(a_1 - a_0) \\ \dots \\ \frac{3}{h_{n-1}}(a_n - a_{n-1}) - \frac{3}{h_{n-2}}(a_{n-1} - a_{n-2}) \\ 3f'(b) - \frac{3}{h_{n-1}}(a_n - a_{n-1}) \end{vmatrix}$$

1.2 Aproximări în sensul CMMP

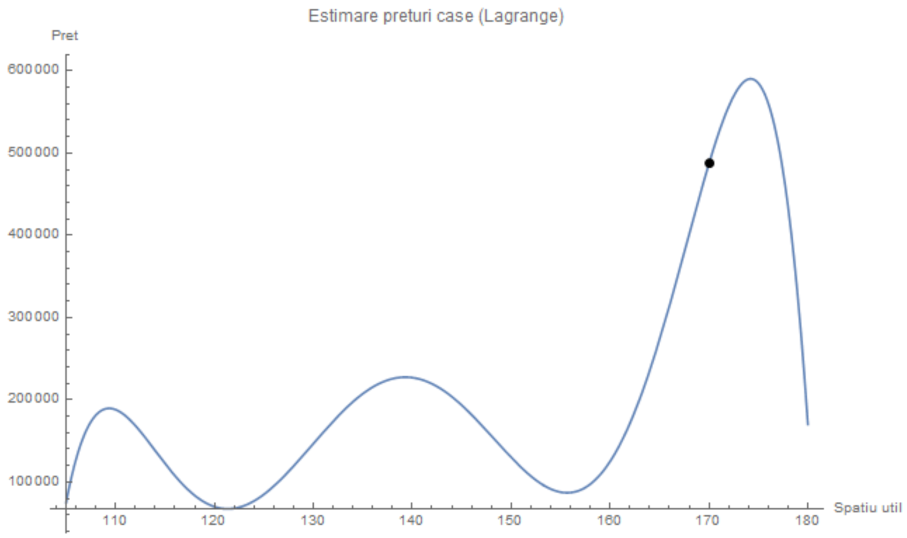
Ideea aproximarii in sensul celor mai mici patrate a pornit de la constatarea faptului ca pentru a aproxima un anumit atribut, nu este neaparat necesara gasirea unui polinom sau a mai multora care trec prin toate punctele cunoscute, ci uneori ar fi mai folositoare gasirea unui polinom care sa treaca printre toate punctele cunoscute, astfel incat datele cunoscute nu sunt intotdeauna exacte sau pur si simplu pentru a micsora erorile de calcul, ne-am dori ceva aproximativ in loc de ceva exact.

Sa presupunem ca am dori sa vindem o casa cu un spatiu util de $170m^2$. Fara sa stim prea multe despre piata si avand experienta doar in domeniul metodelor numerice, decidem sa mergem din vecin in vecin si sa intrebam pentru ce suma si-ar vinde casa (nimic ciudat pana aici). Dintre toti vecinii, doar sapte au bunavointa sa ne raspunda si adunam urmatoarele informatii:

Spatiu util	105	120	125	126	150	160	180
Pret (EUR)	75000	70000	85000	95000	130000	125000	170000



Folosind ce am invatat pana acum, ne gandim sa interpolam aceste puncte si sa gasim o aproximare in punctul reprezentativ valorii de $170m^2$.



Dupa cum putem observa, aceasta aproximare nu ne ajuta foarte mult si pare sa supraevalueze valoarea reala a casei. Uitandu-ne din nou la graficul anterior care contine estimarile vecinilor realizam ca acestea sunt probabil supuse unor erori subiective si ca ne-ar ajuta mult mai mult o dreapta care trece printre toate punctele, decat un polinom care trece prin ele.

Ne-ar interesa o linie $a_1x + a_0$ care sa treaca printre punctele date si care sa aproximeze functia, bazandu-ne pe faptul ca suportul cunoscut de puncte este considerat foarte apropiat de realitate.

Daca $x_0, x_1, ..., x_n$ sunt $n + 1$ puncte distincte in care functia f este aproximata, atunci ne intereseaza un polinom liniar unic care trece printre toate punctele cunoscute si minimizeaza eroarea:

$$E \equiv E(a_1, a_0) = \sum_{i=0}^{n}[f(x_i) - (a_1x + a_0)]^2.$$

Acest polinom se numeste aproximarea liniara in sensul celor mai mici patrate a functiei f cunoscuta in punctele $x_0, x_1, ..., x_n$.

Pentru a minimiza eroarea trebuie ca derivatele partiale ale functiei eroare in a_0 si a_1 sa fie egale cu 0.

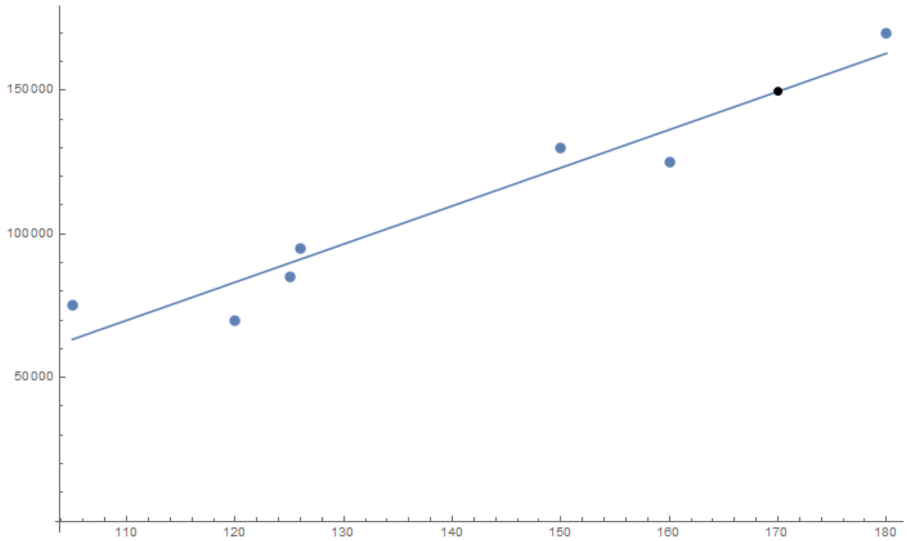
$\frac{\partial}{\partial a_0}\sum\limits_{i=0}^{n}[f(x_i) - (a_1 + a_0)]^2 = 2\sum\limits_{i=0}^{n}(f(x_i) - a_1x_i - a_0)(-1)$

$\frac{\partial}{\partial a_1}\sum\limits_{i=0}^{n}[f(x_i) - (a_1 + a_0)]^2 = 2\sum\limits_{i=0}^{n}(f(x_i) - a_1x_i - a_0)(-x_i)$

$\Rightarrow a_0 = \frac{\sum_{i=0}^{n} x_i^2 \sum_{i=0}^{n} f(x_i) - \sum_{i=0}^{n} x_iy_i \sum_{i=0}^{n} x_i}{m\sum_{i=0}^{n} x_i^2 - (\sum_{i=0}^{n} x_i)^2}$

$\Rightarrow a_1 = \frac{m\sum_{i=0}^{n} x_iy_i - \sum_{i=0}^{n} x_i \sum_{i=0}^{n} y_i}{m\sum_{i=0}^{n} x_i^2 - (\sum_{i=0}^{n} x_i)^2}$

Astfel, pentru punctele noastre coeficientii rezultati vor fi: $a_0 = -7.6165 * 10^4$ si $a_1 = 1328.3$. Acestia vor aproxima un pret in jurul valorii de 149646 EUR, care este mult mai apropiat de valoarea reala a casei.


150000
100000
50000
110
120
130
140
150
160
170
180

Capitolul 2

UEFA

Dupa ce am aflat cateva modalitati de aproximare, ne propunem sa analizam sportul care la nivel mondial are cei mai multi fani, aproximativ 3.5 miliarde: fotbalul. Am urmarit parcursul catorva echipe in Liga Campionilor pe care le-am considerat cu sanse reale la castigarea trofeului, in cativa ani consecutivi, si am incercat sa le prezicem o parte dintre statistici. Rezultatul ne-a surprins si in urmatorul capitol il vom impartasi cu cititorii.

Pentru fiecare rezultat descoperit vom afisa, pe langa grafice, si codul scris in **Wolfram Mathematica** care a generat graficele. Astfel, ne asteptam ca cititorii sa descopere si modul in care se prelucreaza datele in domeniul metodelor numerice, nu numai rezultatele finale.

Pentru echipele care nu au ajuns in finala vom folosi statisticile de la toate meciurile in afara de primul sau al doilea de dupa faza grupelor, pe care ne propunem sa il aproximam folosind toate celelalte meciuri jucate in decursul anului corespunzator (chiar si dupa acesta). Posesia, numarul de pase complete sau numarul de faulturi pentru care vom calcula numarul total de suturi spre poarta vor fi cele reale, din meciurile alese.

Pentru cele care au trecut de faza semifinalelor vom incerca sa aproximam meciul din finala folosind datele obtinute din toate meciurile jucate pana atunci. Consideram valoarea atributului cautat ca fiind media valorilor de pe parcurs. De exemplu, vom considera media posesiilor de pe parcurs ca fiind posesia din finala.

Initial, vom construi suportul interpolarii din statisticile tuturor meciurilor in afara de cel pe care dorim sa il aproximam, sub forma unor perechi de puncte ordonate dupa abscisa.

Cu ajutorul suportului interpolarii vom observa evolutiile interpolarilor si functiilor de aproximare despre care am vorbit in partea teoretica.

Primul pas va fi sa observam cel mai simplu tip de interpolare, adica interpolarea liniara. In continuare, vom aproxima cu ajutorul polinomului de interpolare Lagrange. Urmatoare modalitate de aproximare va fi prin functii de tip spline care folosesc functii de imbinare patratice sau cubice. Vom putea observa, de asemenea, ca interpolarea liniara este o interpolare de tip spline care foloseste functii liniare de imbinare. Dupa ce vom fi interpolat cu functii liniare, Lagrange si spline, vom gasi aproximarile liniare si patratice in sensul celor mai mici patrate. Va fi interesant de observat cum aceste aproximari vor trece printre punctele care construiesc suportul interpolarii. Odata ce toate interpolarile si aproximarile propuse vor fi construite, le vom afisa in acelasi grafic, urmand sa il centram in punctul reprezentand valoarea reala pe care o vom cauta, adica posesia, numarul de pase complete sau numarul de faulturi.

Din cauza erorilor de aproximare ale metodelor folosite, uneori vom obtine rezultate valide doar de la anumite aproximari sau interpolari. Astfel, vom elimina modalitatile care nu se incadreaza in limitele impuse.

2.1 Liga Campionilor 2012-2013

Pentru anul 2012-2013 ne-am propus sa urmarim evolutia suturilor spre poarta fata de procentajul posesiei.

Am selectat urmatoarele echipe:

- Borussia Dortmund
- FC Barcelona
- FC Bayern Munich
- Manchester United F.C.

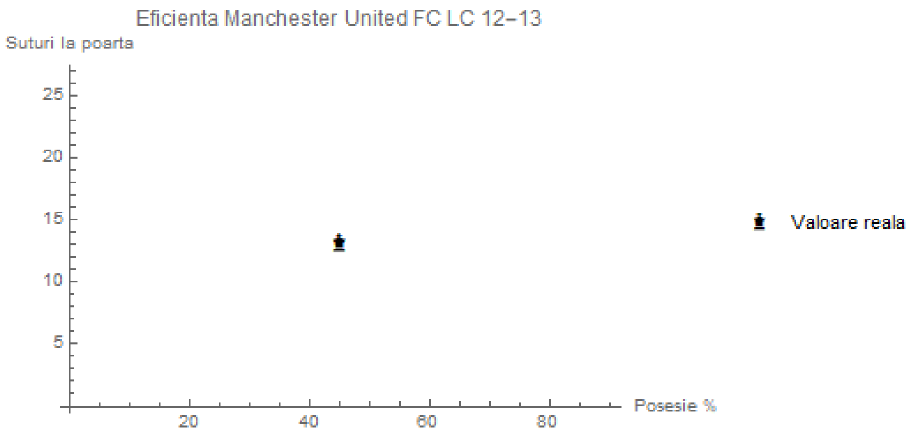
- Real Madrid C.F.

2.1.1 Manchester United F.C.

Ne propunem sa aproximam numarul de suturi spre poarta ale echipei Machester United F.C., raportat la procentajul posesiei din primul meci jucat dupa faza grupelor: Real Madrid C.F. - Manchester United F.C., care s-a incheiat cu scorul de 1-1.

In acest sezon competitional, 2012-2013, Manchester United F.C. a jucat opt meciuri in total, aceasta oprindu-se in faza optimilor.

In graficul de mai jos observam valoarea reala din meciul pe care dorim sa il aproximam, si anume 13 suturi spre poarta avand o posesie de 45%.



Suportul interpolarii

Observam ca accest vector este ordonat dupa primul termen, cel care reprezinta posesia, deoarece acest lucru este necesar pentru a putea aplica metodele enumerate in capitolul 1.

```
suportUnited = {{38, 18}, {48, 18}, {49, 9}, {53, 12}, {57, 9}, {60, 24}, {69, 14}}
```

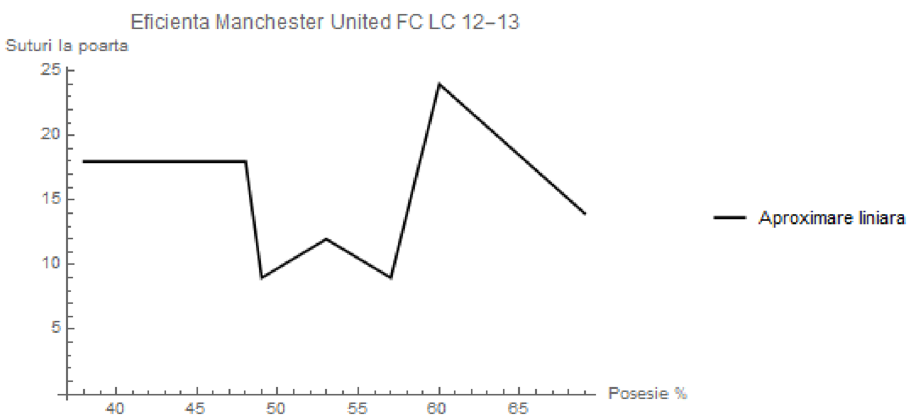
Interpolare liniara

Interpolarea liniara este prima metoda numerica pe care o vom aborda pentru o reprezentare grafica a suportului interpolarii. Aceasta reprezinta trasarea unor segmente intre fiecare doua puncte consecutive.

Putem observa codul Wolfram Mathematica folosit pentru a genera graficul interpolarii liniare mai jos.

```
plotLiniarUnited = ListPlot[
  suportUnited,
  AxesOrigin->{37, 0},
  AxesLabel->{"Posesie %", "Suturi la poarta"},
  PlotLabel->"Eficienta Manchester United F.C. LC 12-13",
  Joined->True,
  PlotStyle->Black,
  PlotLegends->{"Aproximare liniara"}
]
```

Graficul interpolarii liniare arata in modul urmator.



Interpolare Lagrange

Urmeaza interpolarea cu ajutorul polinomului Lagrange care va avea grad maxim 6 deoarece folosim 7 puncte in suportul interpolarii.

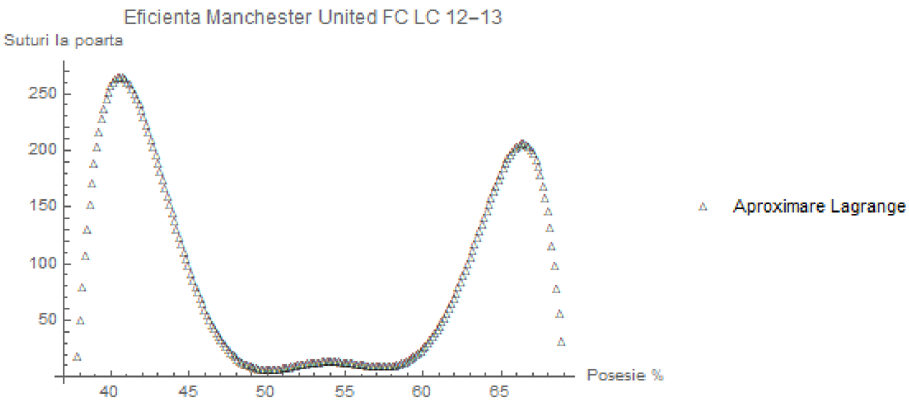
Abordarea in Wolfram Mathematica este urmatoarea.

```
functieLagrangeUnited = InterpolatingPolynomial[
     suportUnited,
     x
]
```

Vom restrange calcularea acestui polinom intre abscisa primului punct, 38, si abscisa ultimului punct, 69, din suportul interpolarii deoarece nu avem date pe care le putem folosi in afara acestui interval.

```
puncteLagrangeUnited = Table[
     functieLagrangeUnited,
     {x, 38, 69, 0.15}
]

plotLagrangeUnited = ListPlot[
     puncteLagrangeUnited,
     DataRange -> {38, 69},
     AxesOrigin -> 37, 0,
     AxesLabel->{"Posesie %", "Suturi la poarta"},
     PlotLabel->"Eficienta Manchester United F.C. LC 12-13",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare Lagrange"},
     PlotMarkers -> {"\[EmptyUpTriangle]"}
]
```

Graficul interpolarii Lagrange este urmatorul.



Interpolari de tip spline

Urmeaza interpolarile cu functii de imbinare de tip spline. Vom incepe cu functii spline patratice si cu modul in care se pot genera aceaste interpolari in Wolfram Mathematica.

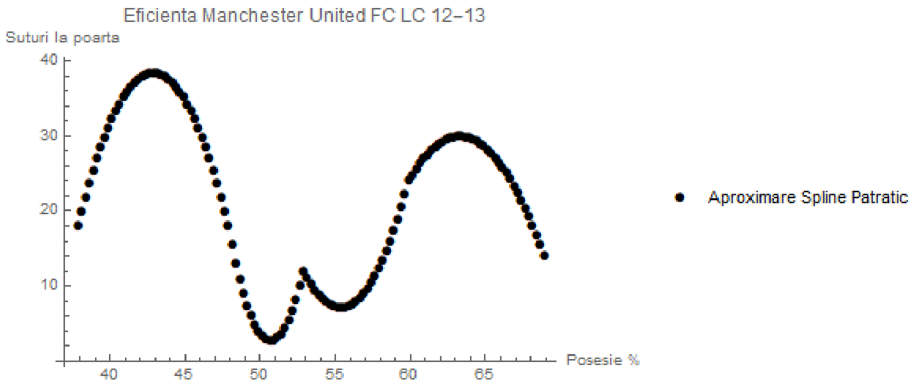
```
functieSplinePatraticUnited = Interpolation[
     suportUnited,
     InterpolationOrder->2][x]
]

puncteSplinePatraticUnited = Table[
     functieSplinePatraticUnited,
     {x, 38, 69, 0.25}
]
```

Generarea graficului este asemanatoare cu cea pentru interpolarea cu polinom Lagrange.

```
plotSplinePatraticUnited = ListPlot[
     puncteSplinePatraticUnited,
```

```
    DataRange -> {38, 69},
    AxesOrigin -> {37, 0},
    AxesLabel->{"Posesie %", "Suturi la poarta"},
    PlotLabel->"Eficienta Manchester United F.C. LC 12-13",
    PlotStyle->Black,
    PlotLegends -> {"Aproximare Spline Patratic"},
    PlotMarkers -> {"\[FilledCircle]"}
]
```

Graficul interpolarii spline cu ajutorul functiilor de imbinare patratice arata astfel.

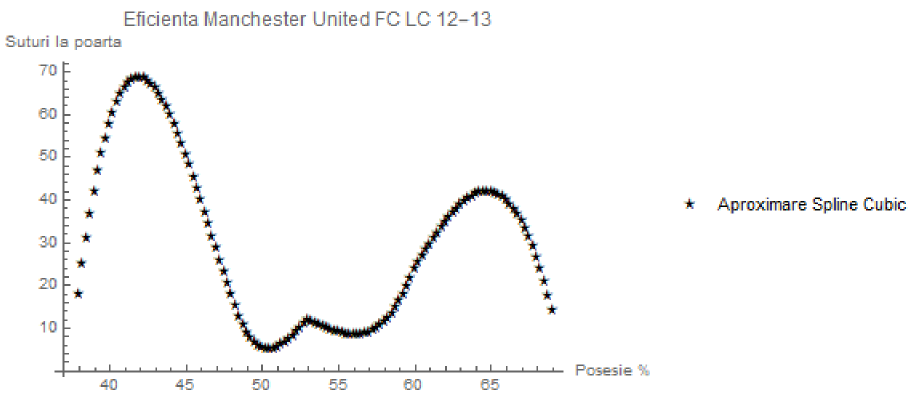


Observam ca spre deosebire de interpolarea cu ajutorul polinomului Lagrange, valorile din capetele suportului de interpolare sunt mult mai apropiate de niste valori posibil reale (40 de suturi spre poarta in loc de 250). Pentru spline-uri cu functii cubice, procedeul este asemanator cu cel folosit pentru functiile patratice.

```
functieSplineCubicUnited = Interpolation[
    suportUnited,
    InterpolationOrder->3][x]
]
```

```
   puncteSplineCubicUnited = Table[
     functieSplineCubicUnited,
     {x, 38, 69, 0.25}
]

plotSplineCubicUnited = ListPlot[
     puncteSplineCubicUnited,
     DataRange -> {38, 69},
     AxesOrigin -> {37, 0},
     AxesLabel->{"Posesie %", "Suturi la poarta"},
     PlotLabel->"Eficienta Manchester United F.C. LC 12-13",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare Spline Cubic"},
     PlotMarkers -> {"\[FivePointedStar]"}
]
```

Se observa ca graficul interpolarii spline cu functii de imbinare cubice oscileaza mai mult decat cel care foloseste functii patratice.

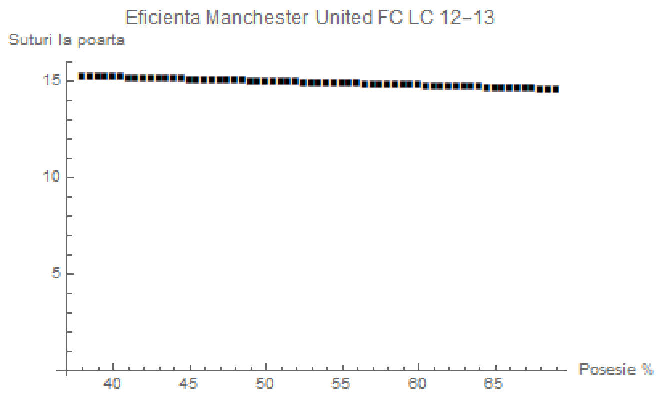


Aproximari in sensul CMMP

In continuare vom aproxima o functie liniara care trece printre punctele din suportul interpolarii in sensul celor mai mici patrate.

```
functieCMMPLiniarUnited = Fit[
     suportUnited,
     {1, x},
     x
]

puncteCMMPLiniarUnited = Table[
     functieCMMPLiniarUnited,
     {x, 38, 69, 0.4}
]

plotCMMPLiniarUnited = ListPlot[
     puncteCMMPLiniarUnited,
     DataRange -> {38, 69},
     AxesOrigin -> {37, 0},
     AxesLabel->{"Posesie %", "Suturi la poarta"},
     PlotLabel->"Eficienta Manchester United F.C. LC 12-13",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare CMMP Liniar"},
     PlotMarkers -> {"\[FilledSquare]"}
]
```

Graficul aproximarii liniare in sensul celor mai mici patrate arata in felul urmator si vom compara putin mai tarziu cu interpolarea liniara ca sa observam pozitia relativa a punctelor aproximate fata de cele din suportul interpolarii.

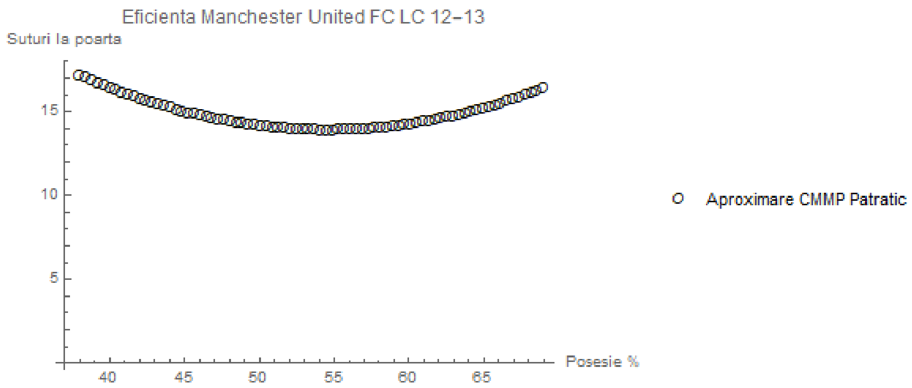


De data aceasta, vom alege un polinom patratic care sa treaca printre toate punctele pentru o eventuala aproximare mai buna.

```
functieCMMPPatraticUnited = Fit[
     suportUnited,
     {1, x, x^2},
     x
]

puncteCMMPPatraticUnited = Table[
     functieCMMPPatraticUnited,
     {x, 38, 69, 0.4}
]

plotCMMPPatraticUnited = ListPlot[
     puncteCMMPPatraticUnited,
     DataRange -> {38, 69},
     AxesOrigin -> {37, 0},
     AxesLabel->{"Posesie %", "Suturi la poarta"},
     PlotLabel->"Eficienta Manchester United F.C. LC 12-13",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare CMMP Patratic"},
```

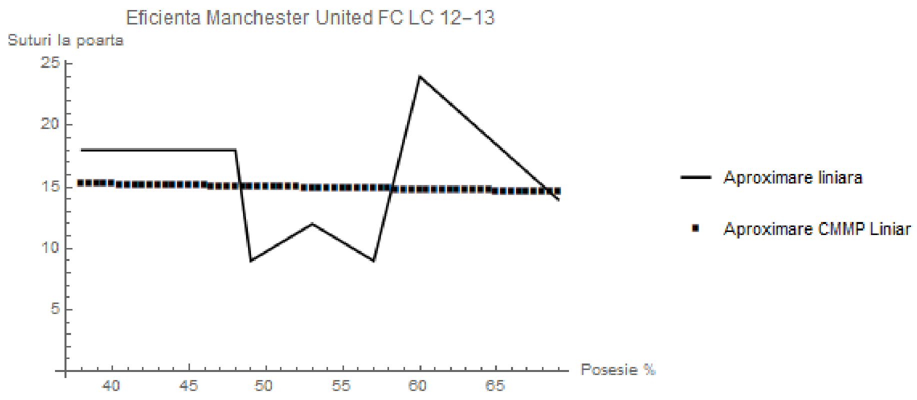
```
    PlotMarkers -> {"O"}
]
```

Graficul aproximarii patratice in sensul celor mai mici patrate este urmatorul.

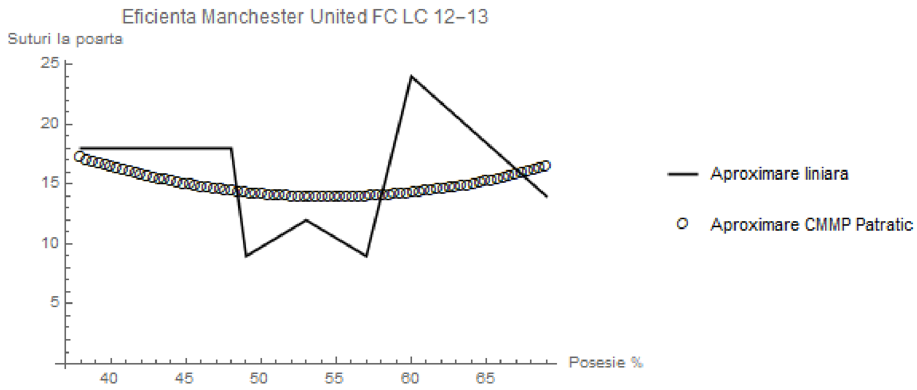


Urmeaza sa afisam cele doua aproximari in sensul celor mai mici patrate raportate la aproximarea liniara pentru sporirea claritatii problemei.

```
Show[plotLiniarUnited, plotCMMPLiniarUnited]
```

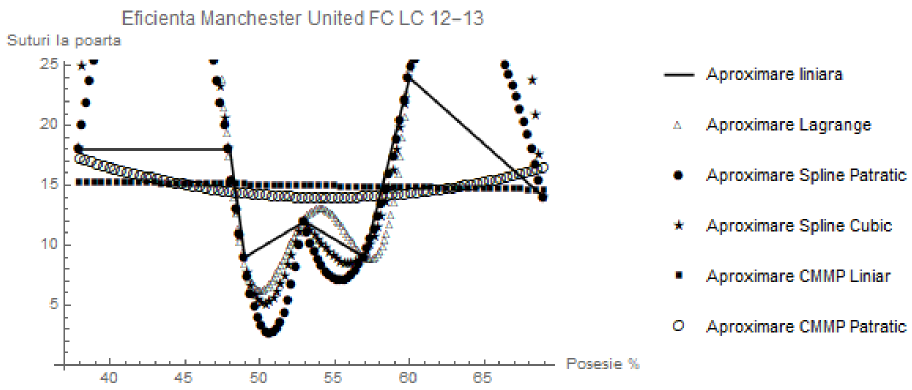


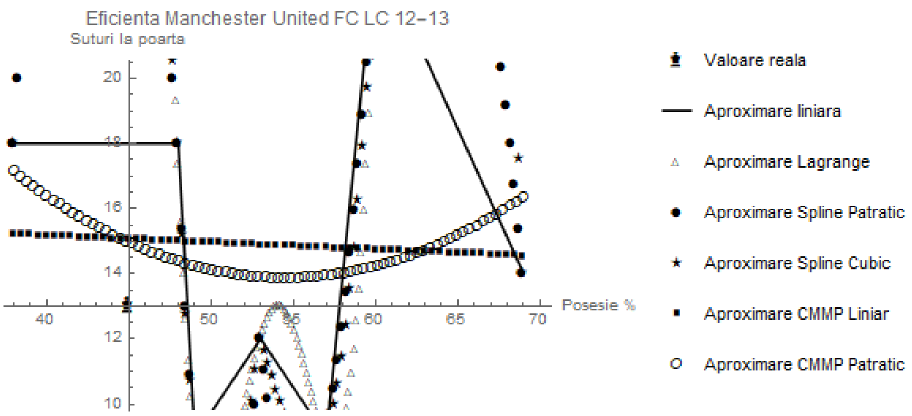
```
Show[plotLiniarUnited, plotCMMPLiniarUnited]
```



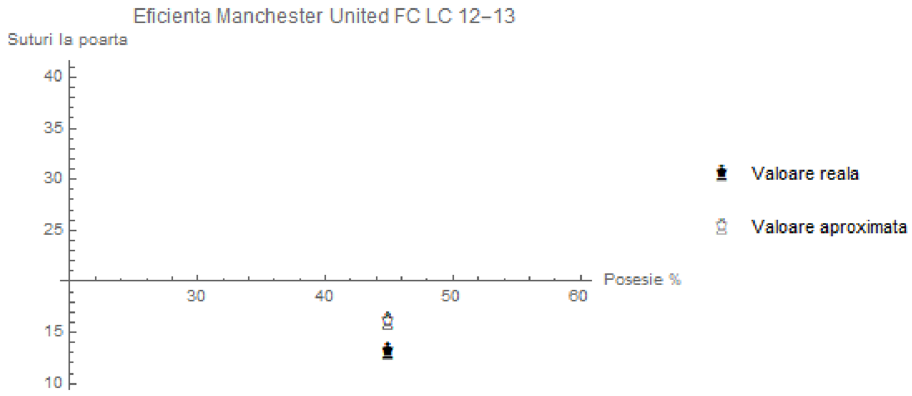
Dupa ce am construit toate interpolarile si aproximarile propuse urmeaza sa le afisam in acelasi grafic, iar apoi sa il centram in punctul reprezentand valoarea reala pe care o cautam ({45, 13}), adica posesia si numarul de suturi spre poarta din primul meci de dupa grupe.

```
Show[
    plotLiniarUnited,
    plotLagrUnited,
    plotSplinePatraticUnited,
    plotSplineCubicUnited,
    plotCMMPLiniarUnited,
    plotCMMPPatraticUnited,
    PlotRange -> {{37, 70}, {10, 20}},
    AxesOrigin -> {45, 13}
]
```





Din cauza erorilor de aproximare ale metodelor folosite, vom folosi doar rezulatele obtinute de la aproximarea liniara, aproximarea liniara in sensul celor mai mici patrate si aproximarea patratica in sensul celor mai mici patrate. Astfel, in punctul de coordonata $x = 45$ vom calcula media aritmetica a celor trei aproximari si va rezulta $y = 15.98$. Observam ca distanta dintre valoarea reala si valoarea aproximata este < 2, un rezultat promitator.

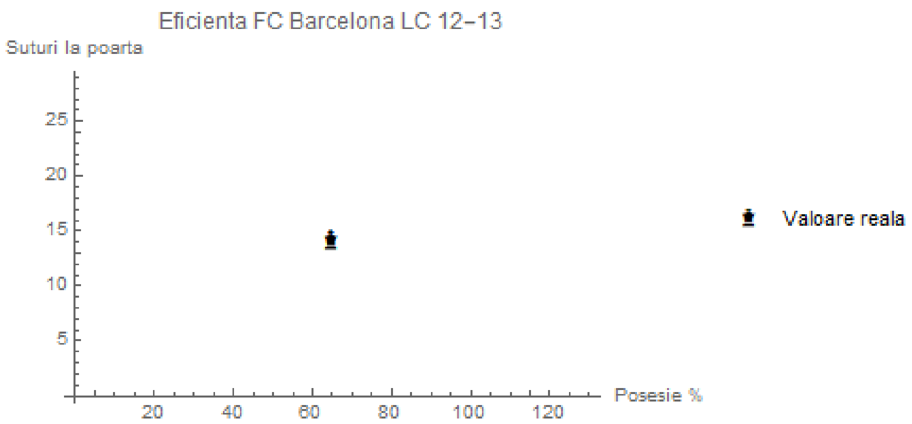


2.1.2 FC Barcelona

Dorim in continuare sa aproximam numarul de suturi spre poarta ale echipei FC Barcelona din al doilea meci de dupa faza grupelor din sezonul 2012-2013 al Ligii Campionilor, respectiv FC Barcelona - A.C. Milan 4 - 0. In acest meci, Barcelona a avut o posesie de 65% si 14 suturi spre poarta.

Ne vom raporta la douasprezece meciuri deoarece Barcelona a fost eliminata in semifinalele competitiei.

Valoarea reala pe care dorim sa o aproximam (65, 14) este reprezentata grafic mai jos.



Suportul interpolarii

Suportul interpolarii va fi format din datele adunate de la unsprezece meciuri, adica unsprezece perechi de tipul (posesie, suturi spre poarta) sortate dupa posesie.

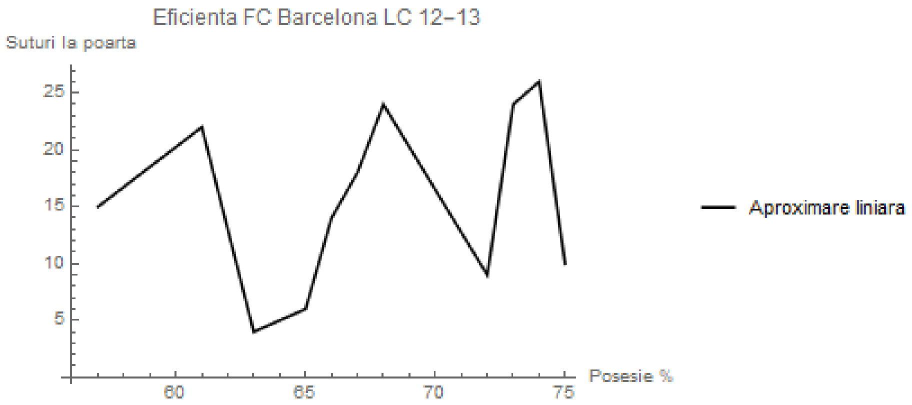
```
suportBarcelona = {{57, 15}, {61, 22}, {63, 4}, {65, 6}, {66,
14}, {67, 18}, {68, 24}, {72, 9}, {73, 24}, {74, 26}, {75, 10}}
```

Interpolare liniara

In continuare dorim sa afisam un grafic reprezentand interpolarea liniara a acestor puncte, aceasta fiind prima modalitate care ne va ajuta sa aproximam numarul de suturi spre poarta din meciul selectat mai sus.

```
plotLiniarBarcelona = ListPlot[
     suportBarcelona,
     AxesOrigin->{56, 0},
     AxesLabel->{"Posesie %", "Suturi la poarta"},
     PlotLabel->"Eficienta FC Barcelona LC 12-13",
     Joined->True,
     PlotStyle->Black,
     PlotLegends->{"Aproximare liniara"}
]
```

Graficul interpolarii liniare este urmatorul.



Interpolare Lagrange

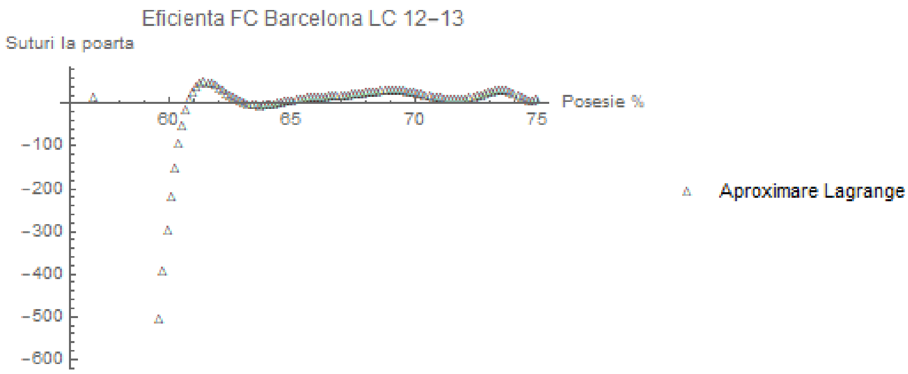
Al doilea tip de interpolare pe care o vom folosi se realizeaza cu ajutorul polinomului Lagrange care va fi de grad maxim 10 deoarece folosim 11 puncte in suportul interpolarii.

Modul in care acesta este construit in Wolfram Mathematica este urmatorul.

```
functieLagrangeBarcelona = InterpolatingPolynomial[
     suportBarcelona,
     x
]

puncteLagrangeBarcelona = Table[
     functieLagrangeBarcelona,
     {x, 57, 75, 0.15}
]

plotLagrangeBarcelona = ListPlot[
     puncteLagrangeBarcelona,
     DataRange -> {57, 75},
     AxesOrigin -> 56, 0,
     AxesLabel->{"Posesie %", "Suturi la poarta"},
```

```
    PlotLabel->"Eficienta FC Barcelona LC 12-13",
    PlotStyle->Black,
    PlotLegends -> {"Aproximare Lagrange"},
    PlotMarkers -> {"\[EmptyUpTriangle]"}
]
```

Graficul interpolarii Lagrange este cel de mai jos.



Observam ca acesta poate avea si valori negative, lucru care nu poate fi valabil din punct de vedere fizic gandindu-ne la ce dorim noi sa aproximam, adica suturile spre poarta. Acest fapt ne arata cat de sensibil este acest polinom la oscilatii (de obicei la capete) si faptul ca trebuie folosit cu grija si de cele mai multe ori folosind cat mai multe date si cat mai apropiate pentru a micsora eroarea.

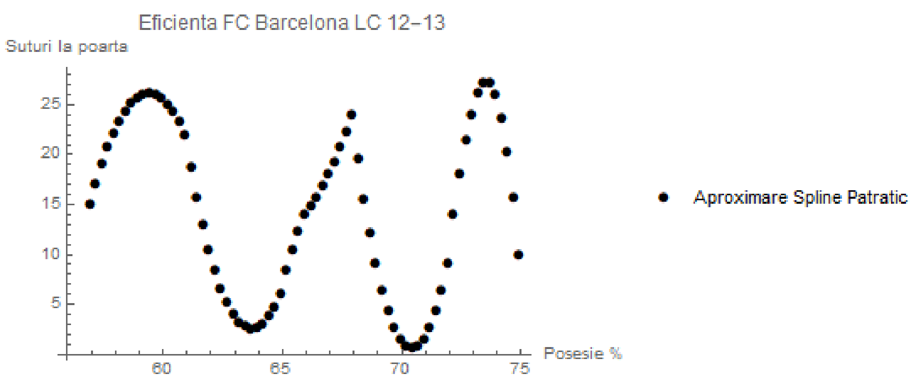
Interpolari de tip spline

Ne axam acum pe interpolarile de tip spline folosind functii de imbinare patratice si cubice. Codul Wolfram Mathematica care ne permite obtinerea acestor interpolari este urmatorul.

```
functieSplinePatraticBarcelona = Interpolation[
    suportBarcelona,
    InterpolationOrder->2][x]
```

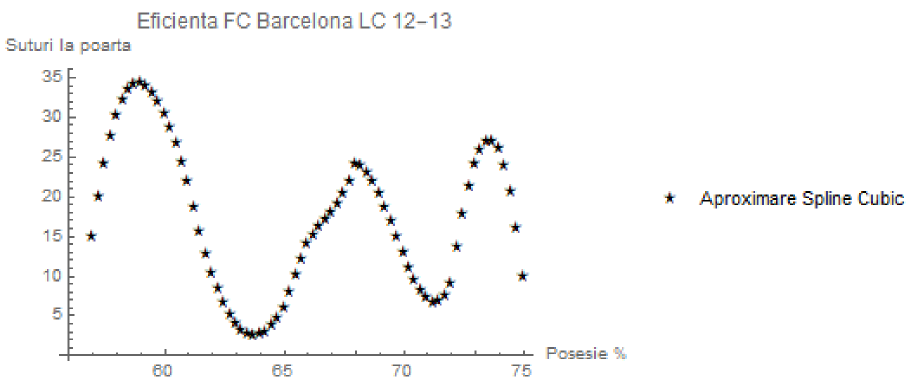
```
]

puncteSplinePatraticBarcelona = Table[
     functieSplinePatraticBarcelona,
     {x, 57, 75, 0.25}
]

plotSplinePatraticBarcelona = ListPlot[
     puncteSplinePatraticBarcelona,
     DataRange -> {57, 75},
     AxesOrigin -> {56, 0},
     AxesLabel->{"Posesie %", "Suturi la poarta"},
     PlotLabel->"Eficienta FC Barcelona LC 12-13",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare Spline Patratic"},
     PlotMarkers -> {"\[FilledCircle]"}
]
```

Graficul interpolarii spline cu functii patratice este cel de mai jos.



Dupa cum se poate observa, acest tip de interpolare este mult mai stabil la capete decat interpolarea Lagrange. Pentru obtinerea interpolarii spline cu functii de imbinare cubice procedeul este analog.

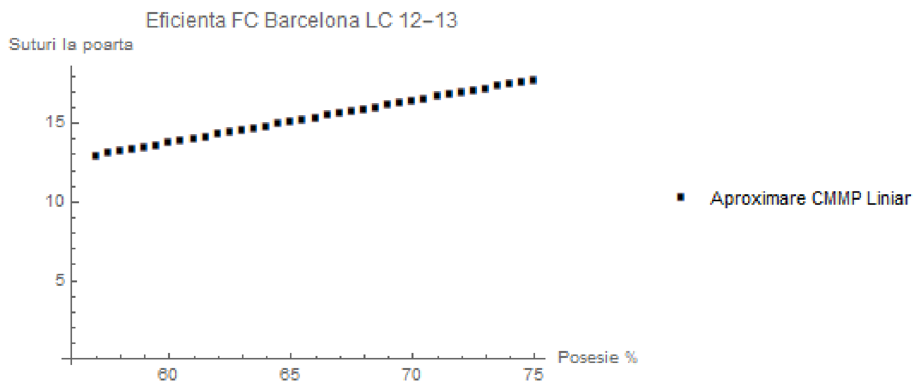
```
functieSplineCubicBarcelona = Interpolation[
     suportBarcelona,
     InterpolationOrder->3][x]
]

puncteSplineCubicBarcelona = Table[
     functieSplineCubicBarcelona,
     {x, 57, 75, 0.25}
]

plotSplineCubicBarcelona = ListPlot[
     puncteSplineCubicBarcelona,
     DataRange -> {57, 75},
     AxesOrigin -> {56, 0},
     AxesLabel->{"Posesie %", "Suturi la poarta"},
     PlotLabel->"Eficienta FC Barcelona LC 12-13",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare Spline Cubic"},
     PlotMarkers -> {"\[FivePointedStar]"}
]
```



Aproximari in sensul CMMP

Ne dorim in continuare sa gasim un polinom liniar care trece printre toate punctele. Pentru a-l gasi pe cel mai bun din punct de vedere al distantei de la punctele din suportul interpolarii pana la el, vom folosi aproximarea in sensul celor mai mici patrate.

```
functieCMMPLiniarBarcelona = Fit[suportBarcelona, {1, x}, x]
puncteCMMPLiniarBarcelona = Table[
     functieCMMPLiniarBarcelona,
     {x, 57, 75, 0.4}
]

plotCMMPLiniarBarcelona = ListPlot[
     puncteCMMPLiniarBarcelona,
     DataRange -> {57, 75},
     AxesOrigin -> {56, 0},
     AxesLabel->{"Posesie %", "Suturi la poarta"},
     PlotLabel->"Eficienta FC Barcelona LC 12-13",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare CMMP Liniar"},
     PlotMarkers -> {"\[FilledSquare]"}
]
```



Observam ca graficul are o panta ascendenta, ceea ce ne arata faptul ca numarul de suturi spre poarta a crescut in acelasi timp cu posesia de-a lungul competitiei pentru FC Barcelona in sezonul 2012-2013.

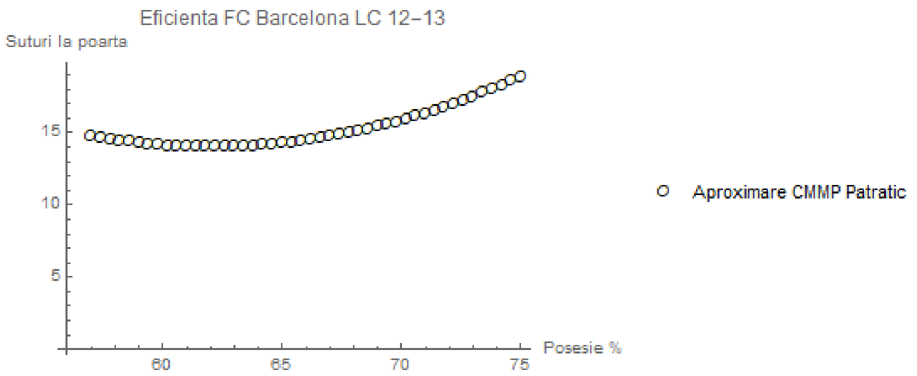
Pentru a calcula aproximarea patratica in sensul celor mai mici patrate putem folosi Wolfram Mathematica astfel.

```
functieCMMPPatraticBarcelona = Fit[suportBarcelona, {1, x, x^2},
x]

puncteCMMPPatraticBarcelona = Table[
     functieCMMPPatraticBarcelona,
     {x, 57, 75, 0.5}
]
```

Restrangem segmentul de date urmarit la multimea [57, 75], reprezentata de primul si ultimul punct din suportul interpolarii, deoarece nu are sens sa incercam aproximam in afara acestora.

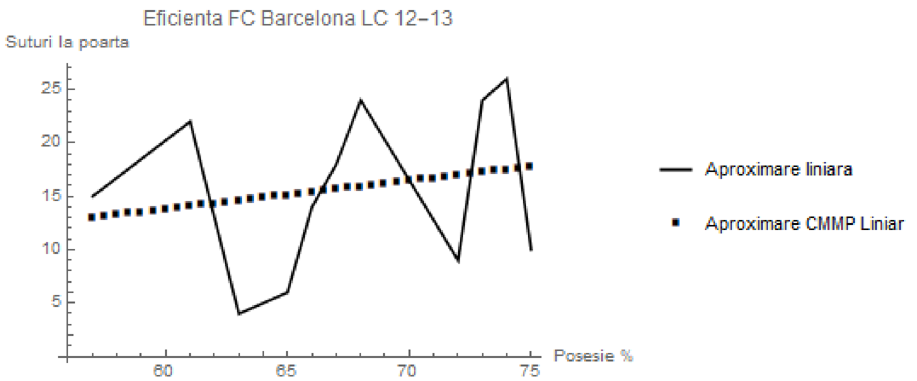
```
plotCMMPPatraticBarcelona = ListPlot[
     puncteCMMPPatraticBarcelona,
     DataRange -> {57, 75},
     AxesOrigin -> {56, 0},
     AxesLabel->{"Posesie %", "Suturi la poarta"},
     PlotLabel->"Eficienta FC Barcelona LC 12-13",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare CMMP Patratic"},
     PlotMarkers -> {"O"}
]
```

Trasarea graficului se face in modul urmator si ne arata prin forma sa avantajele fata de o aproximare liniara. Fiind un polinom patratic, poate minimiza mai mult eroarea distantei de la el la punctele din suportul de interpolare.



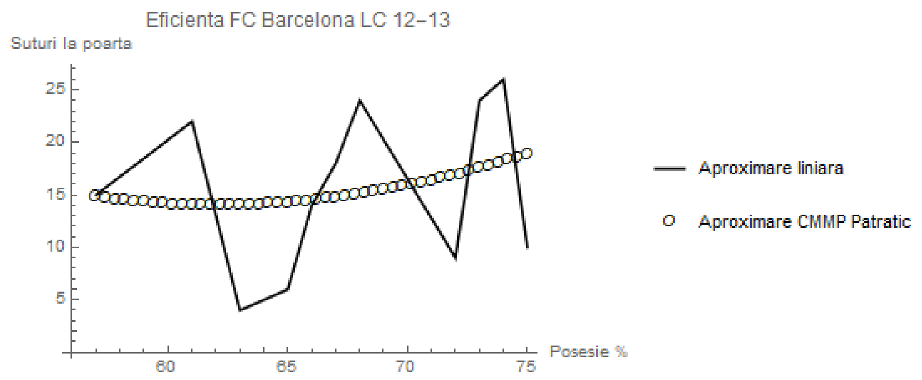
In continuare, vom afisa interpolarea liniara raportata la aproximarile in sensul celor mai mici patrate pentru a observa cum acestea trec perfect printre punctele obtinute din celelalte unsprezece meciuri.

```
Show[plotLiniarBarcelona, plotCMMPLiniarBarcelona]
```



Observam ca avand deja create in Wolfram Mathematica variabilele de plotare a celor doua modalitati de interpolare, respectiv aproximare, vom putea foarte usor sa le imbinam in acelasi grafic prin utilizarea functiei "Show".

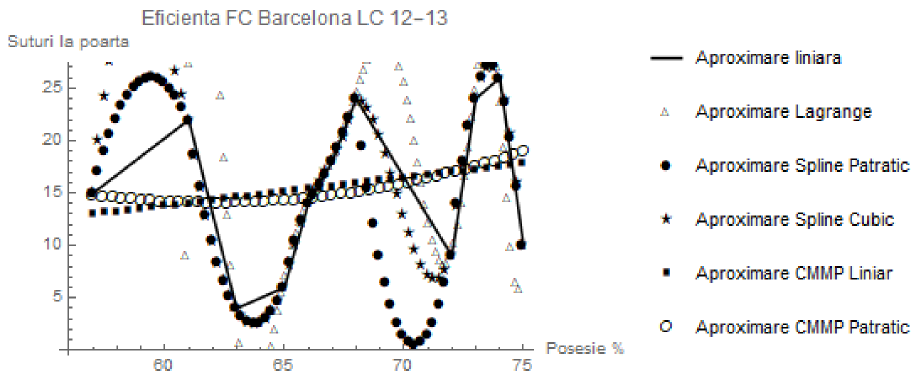
```
Show[plotLiniarBarcelona, plotCMMPPatraticBarcelona]
```



Centram in punctul reprezentand valoarea reala pe care o cautam ({65, 14}), adica posesia si numarul de suturi la poarta din al doilea meci de dupa grupe.

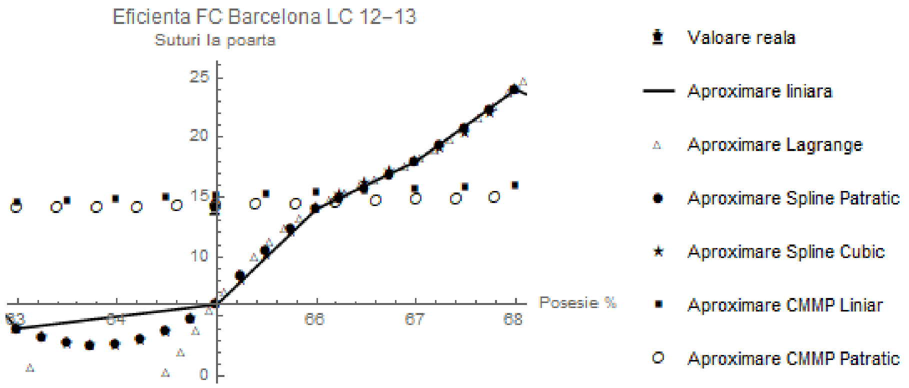
```
Show[
    plotLiniarBarcelona,
    plotLagrBarcelona,
    plotSplinePatraticBarcelona,
    plotSplineCubicBarcelona,
    plotCMMPLiniarBarcelona,
    plotCMMPPatraticBarcelona,
    PlotRange -> {{63, 68}, {0, 25}},
    AxesOrigin -> {65, 14}
]
```

Afisam si graficul care rezulta din suprapunerea tuturor metodelor de aproximare si interpolare pe care le-am folosit. Putem sa ne orientam folosind acest grafic pentru a folosi sau nu toate rezultatele in aproximarea finala. Daca exista metode care sunt mult prea departe de valoarea reala fiindca oscileaza sau nu sunt potrivite pe setul de puncte disponibil, le vom elimina.

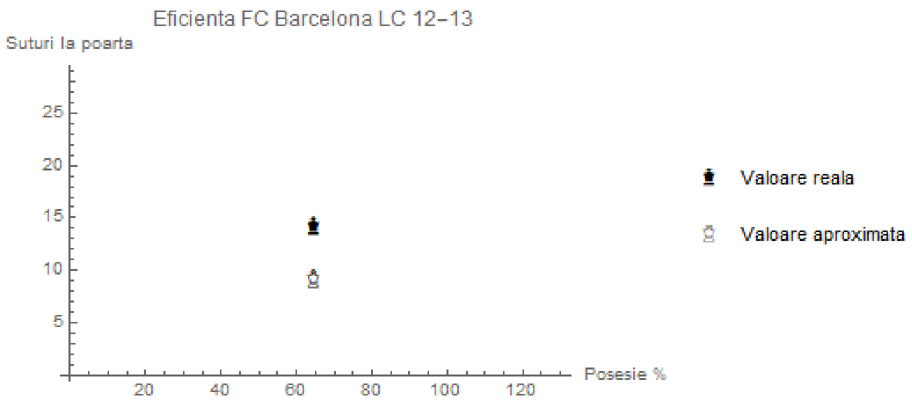


Vom folosi toate rezulatele obtinute si vom obtine, astfel, in punctul de coordonata $x = 65$, adica valoarea reala a posesiei din meciul care ne intereseaza, $y = 9$, reprezentand valoarea aproximata.

Observam ca distanta dintre valoarea reala si valoarea aproximata este egala cu 5. Cele doua aproximari care indeparteaza valoarea aproximata de cea reala sunt aproximarile in sensul celor mai mici patrate, liniara si patratica. Acesta intamplare se poate datora faptului ca suportul interpolarii contine o aglomerare de puncte in jurul punctului reprezentativ pentru posesie 65, astfel avantajand metodele de interpolare in fata celor de aproximare, primele creandu-si o vecinatate de puncte mai mare in jurul valorii pe care noi o aproximam.



Cele doua valori, cea reala si cea aproximata, arata ca mai jos.

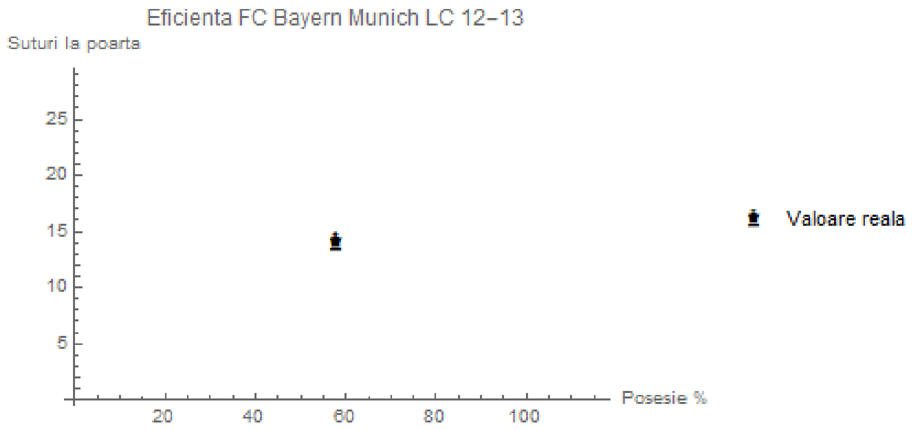


2.1.3 FC Bayern Munich

Acum ne propunem sa aproximam numarul de suturi spre poarta raportate la posesie pentru echipa FC Bayern Munich. Aceasta a fost finalista, jucand astfel treisprezece meciuri in total in sezonul competitional 2012-2013 al Ligii Campionilor. Vom folosi datele de la primele douasprezece meciuri (pana la semifinale, inclusiv acestea) si vom incerca sa aproximam statistica dorita din meciul final, Borussia Dortmund - FC Bayern Munich 1 - 2, unde bavarezii au avut o posesie de 58% si 14 suturi spre poarta.

Spre deosebire de cum procedam pentru echipele care nu au ajuns in finala cand cunosteam posesia meciului pentru care doream sa aproximam numarul de suturi spre poarta, de data aceasta vom considera posesia din meciul final ca fiind media tuturor posesiilor cunoscute din celelalte meciuri.

Valoarea reala, ({58, 14}), arata ca in graficul urmator.



Suportul interpolarii

Acesta este cel mai numeros suport al interpolarii cu care am avut de-a face pana acum. Un numar mai mare de puncte ar trebui, in mod normal dar nu intotdeauna, sa insemne aproximari mai exacte.

```
suportBayern = {{37, 13}, {43, 8}, {45, 16}, {48, 7}, {54, 21},
{55, 21}, {57, 17}, {61, 17}, {62, 18}, {63, 16}, {64, 17},
{67, 17}}
```

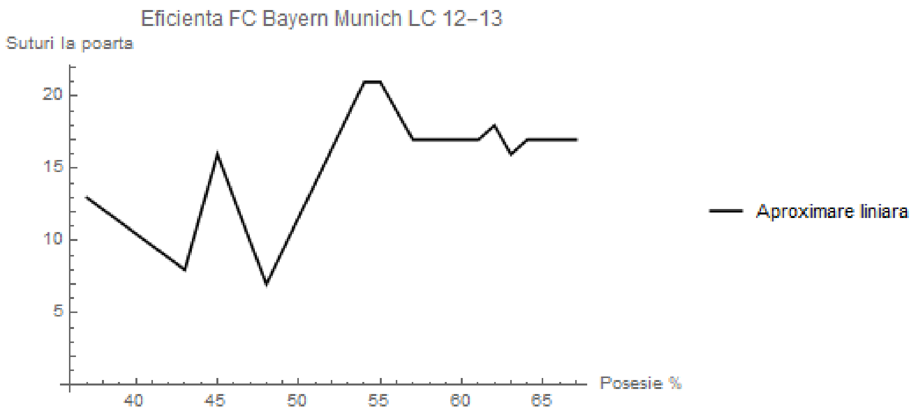
Interpolare liniara

Vom incepe cu interpolarea liniara si cum se poate genera aceasta in Wolfram Mathematica.

```
plotLiniarBayern = ListPlot[
     suportBayern,
     AxesOrigin->{36, 0},
     AxesLabel->{"Posesie %", "Suturi la poarta"},
     PlotLabel->"Eficienta FC Bayern Munich LC 12-13",
     Joined->True,
     PlotStyle->Black,
```

```
     PlotLegends->{"Aproximare liniara"}
]
```

Graficul interpolarii liniare este urmatorul.



Interpolare Lagrange

Urmeaza interpolarea cu polinom Lagrange care va avea grad maxim 11, deoarece sunt 12 puncte in suportul interpolarii. Codul Wolfram Mathematica pentru generare este urmatorul.

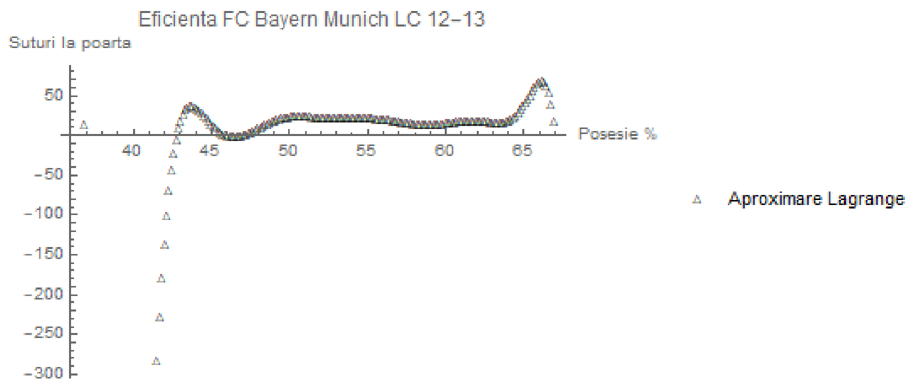
```
functieLagrangeBayern = InterpolatingPolynomial[
     suportBayern,
     x
]
```

Am folosit 37 si 67 ca puncte de margine pentru calcularea polinomului deoarece nimic din ce este in afara acestui interval nu ne intereseaza.

```
puncteLagrangeBayern = Table[
     functieLagrangeBayern,
     {x, 37, 67, 0.15}
]
```

```
plotLagrangeBayern = ListPlot[
     puncteLagrangeBayern,
     DataRange -> {37, 67},
     AxesOrigin -> 36, 0,
     AxesLabel->{"Posesie %", "Suturi la poarta"},
     PlotLabel->"Eficienta FC Bayern Munich LC 12-13",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare Lagrange"},
     PlotMarkers -> {"\[EmptyUpTriangle]"}
]
```

Mai jos este prezentat graficul interpolarii Lagrange pentru FC Bayern Munich in anul competitional 2012-2013. Putem observa si aici valori negative datorita oscilatiilor mari din capete produse de aceasta metoda de interpolare.

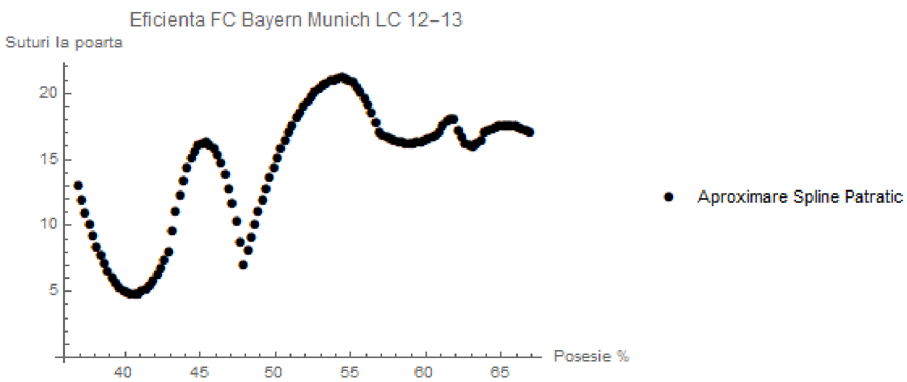


Interpolari de tip spline

In continuare, vom folosi interpolari de tip spline cu functii de imbinare polinoame de grad 2 si 3. Pentru polinoame patratice, procedeul este urmatorul.

```
functieSplinePatraticBayern = Interpolation[     suportBayern,
     InterpolationOrder->2][x]
]

puncteSplinePatraticBayern = Table[
     functieSplinePatraticBayern,
     {x, 37, 67, 0.25}
]

plotSplinePatraticBayern = ListPlot[
     puncteSplinePatraticBayern,
     DataRange -> {37, 67},
     AxesOrigin -> {36, 0},
     AxesLabel->{"Posesie %", "Suturi la poarta"},
     PlotLabel->"Eficienta FC Bayern Munich LC 12-13",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare Spline Patratic"},
     PlotMarkers -> {"\[FilledCircle]"}
]
```

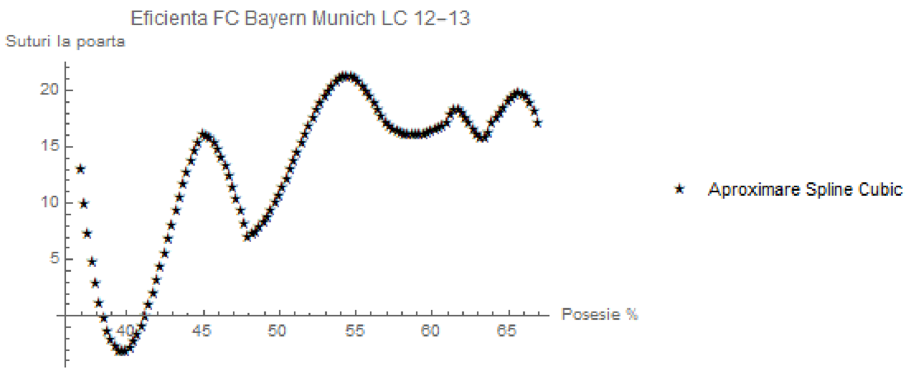
Mai jos se poate observa graficul care reprezinta interpolarea spline cu functii patratice de imbinare.



Pentru abordarea cu functii cubice de imbinare, algoritmul este la fel ca la cele patratice.

```
functieSplineCubicBayern = Interpolation[
     suportBayern,
     InterpolationOrder->3][x]
puncteSplineCubicBayern = Table[
     functieSplineCubicBayern,
     {x, 37, 67, 0.25}
]

plotSplineCubicBayern = ListPlot[
     puncteSplineCubicBayern,
     DataRange -> {37, 67},
     AxesOrigin -> {36, 0},
     AxesLabel->{"Posesie %", "Suturi la poarta"},
     PlotLabel->"Eficienta FC Bayern Munich LC 12-13",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare Spline Cubic"},
     PlotMarkers -> {"\[FivePointedStar]"}
]
```

Graficul interpolarii spline cu functii de imbinare cubice este urmatorul.



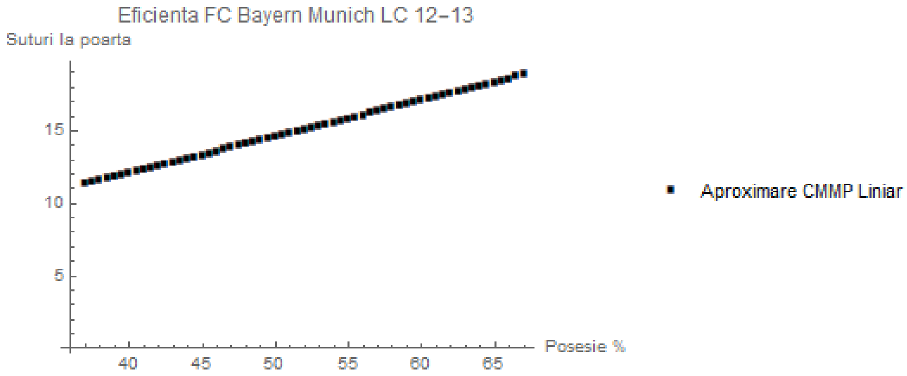
Aproximari in sensul CMMP

Urmeaza aproximarile in sensul celor mai mici patrate. Vom incepe, dupa cum ne-am obisnuit, cu aproximarea liniara si cu modul in care aceasta se genereaza in Wolfram Mathematica. Apoi vom continua cu aproximarea patratica si cu comparatii ale acestor doua aproximari cu interpolarea liniara.

Se observa folosirea functiei ”Fit” pentru gasirea acestor aproximari care, dupa cum ii spune si numele, va incerca sa potriveasca mai multe puncte intr-o linie astfel incat departarea lor fata de punctele reale sa fie minima.

```
functieCMMPLiniarBayern = Fit[suportBayern, {1, x}, x]

puncteCMMPLiniarBayern = Table[
     functieCMMPLiniarBayern,
     {x, 37, 67, 0.4}
]

plotCMMPLiniarBayern = ListPlot[
     puncteCMMPLiniarBayern,
     DataRange -> {37, 67},
     AxesOrigin -> {36, 0},
     AxesLabel->{"Posesie %", "Suturi la poarta"},
     PlotLabel->"Eficienta FC Bayern Munich LC 12-13",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare CMMP Liniar"},
     PlotMarkers -> {"\[FilledSquare]"}
]
```

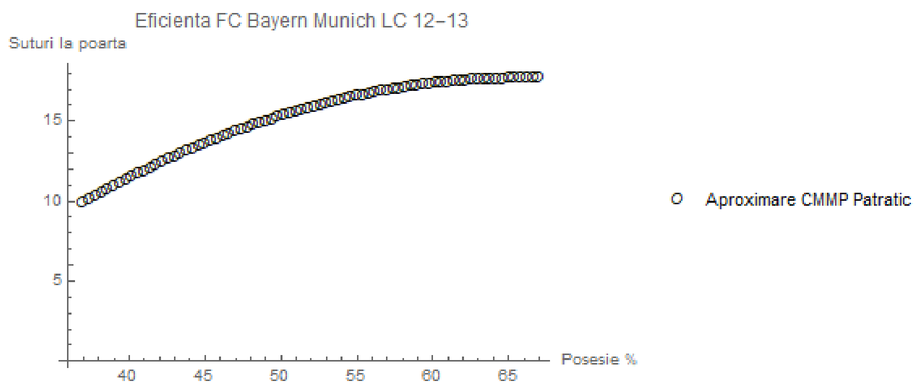
In graficul urmator se poate observa o crestere, ceea ce inseamna ca posesia chiar a influentat liniar numarul de suturi spre poarta ale echipei FC Bayern Munich in sezonul competitional 2012 - 2013.



Aproximarea patratica in sensul celor mai mici patrate se face aproape identic cu cea liniara, dar ne asteptam ca in cazul unei modificari amortizate (cu convexitate, astfel incat sa isi modifice starea dar sa tinda la a reveni la cea initiala) a punctelor din suportul interpolarii, sa aproximeze mai bine si sa arate o curba mai apropiata de ce ne dorim noi.

```
functieCMMPPatraticBayern = Fit[suportBayern, {1, x, x^2}, x]
puncteCMMPPatraticBayern = Table[
     functieCMMPPatraticBayern,
     {x, 37, 67, 0.4}
]

plotCMMPPatraticBayern = ListPlot[
     puncteCMMPPatraticBayern,
     DataRange -> {37, 67},
     AxesOrigin -> {36, 0},
     AxesLabel->{"Posesie %", "Suturi la poarta"},
     PlotLabel->"Eficienta FC Bayern Munich LC 12-13",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare CMMP Patratic"},
     PlotMarkers -> {"O"}
]
```

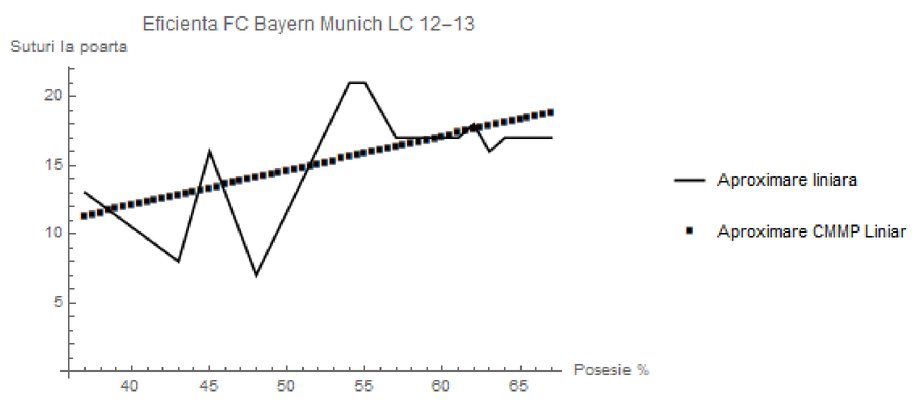
Graficul acestei aproximari arata ca in figura de mai jos.



Acum vom arata diferenta dintre aproximarea liniara in sensul celor mai mici patrate care nu trece prin ci printre punctele din suportul interpolarii si interpolarea liniara care va trece exact prin acele puncte.

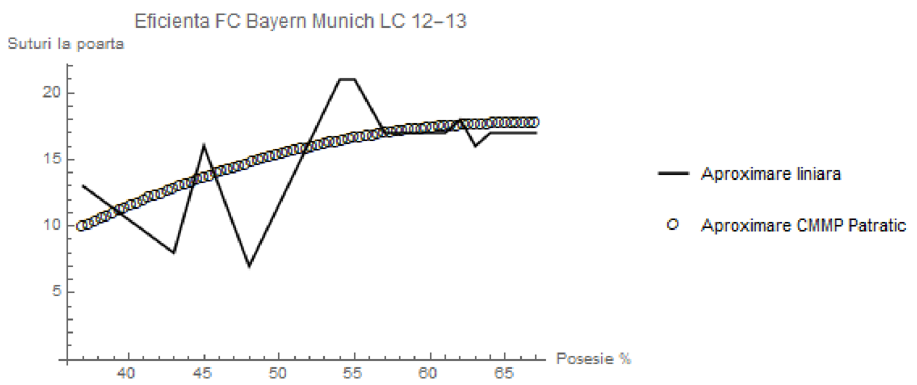
Deoarece cele doua aproximari sunt deja calculate, le putem folosi plotarile pentru a le suprapune.

```
Show[plotLiniarBayern, plotCMMPLiniarBayern]
```



Analog, observam cum punctele din aproximarea patratica in sensul celor mai mici patrate trec exact printre punctele din suportul interpolarii.

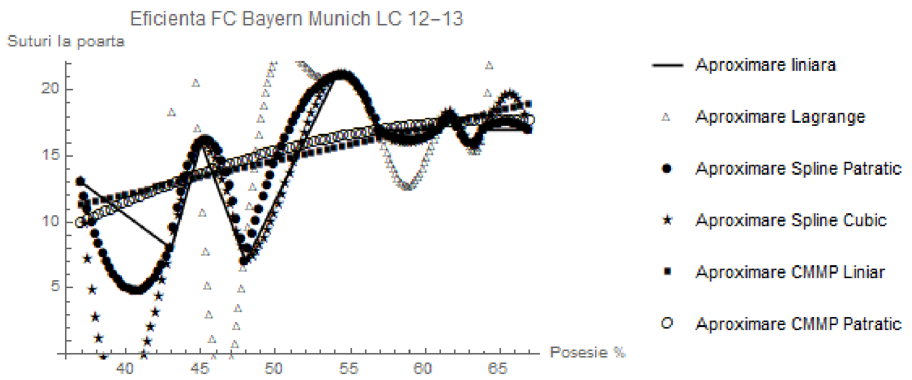
```
Show[plotLiniarBayern, plotCMMPLiniarBayern]
```

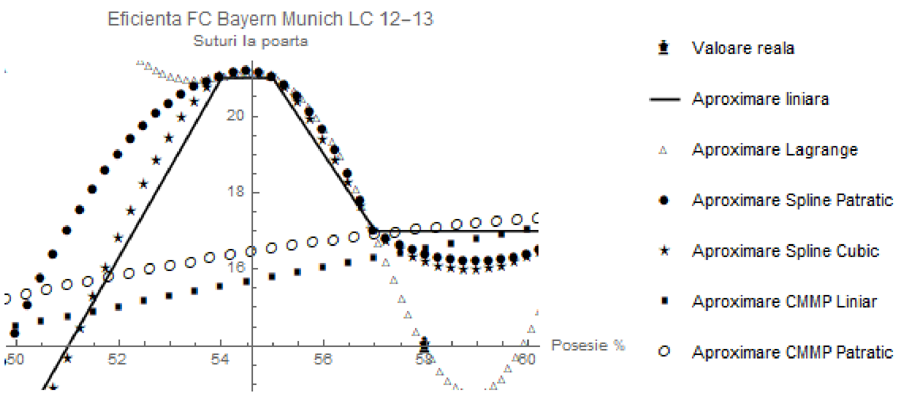


Dupa ce am construit toate interpolarile si aproximarile propuse urmeaza sa le afisam in acelasi grafic, iar apoi sa il centram in punctul reprezentand valoarea reala pe care o cautam ($\{58, 14\}$), adica posesia si numarul de suturi la poarta din finala, pentru a observa mai usor diferentele si eroarea fiecarei metode aplicate.

Folosim toate variabilele create pentru plotare si doar setam marginile graficului, pe ambele axe, Wolfram Mathematica ajutandu-ne cu o sintaxa facila in acest sens.

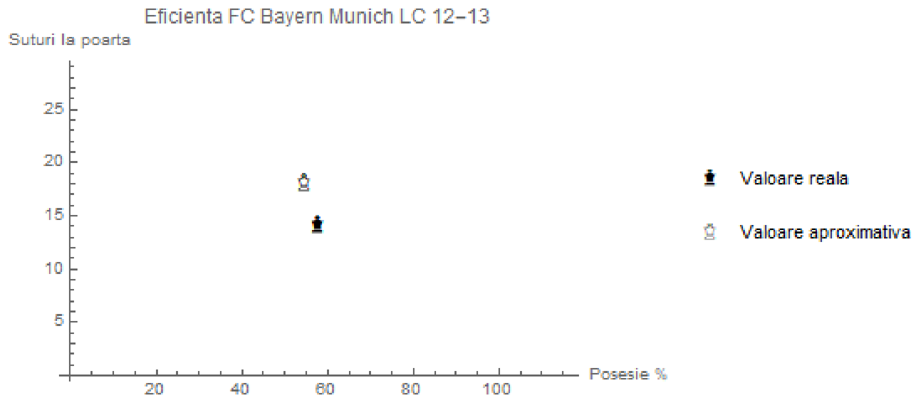
```
Show[
     plotLiniarBayern,
     plotLagrBayern,
     plotSplinePatraticBayern,
     plotSplineCubicBayern,
     plotCMMPLiniarBayern,
     plotCMMPPatraticBayern,
     PlotRange -> {{50, 60}, {13, 21}},
     AxesOrigin -> {58, 14}
]
```





Vom folosi toate rezulatele obtinute si, astfel, in punctul de coordonata $x = 54.6$, adica media absciselor (posesiilor) din suportul interpolarii, vom aproxima $y = 18$.
Observam ca distanta dintre valoarea reala si valoarea aproximata este egala cu 4.

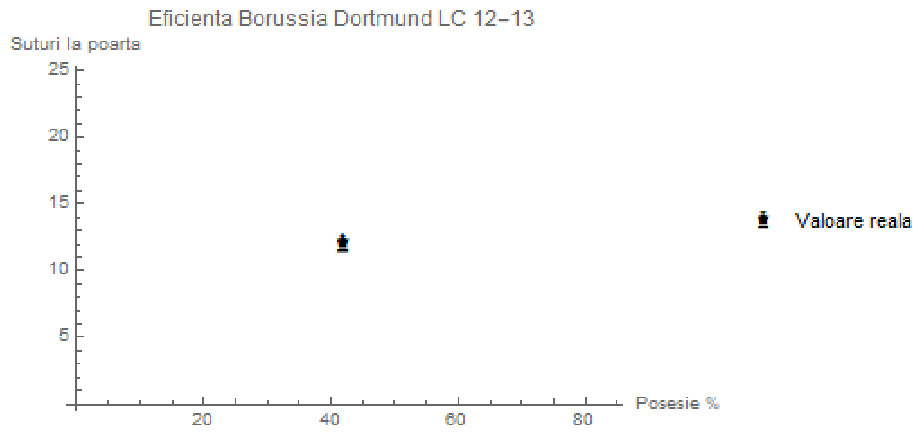
Afisam si graficul cu cele doua valori si vom observa ca spre deosebire de celelalte echipe de pana acum, la Bayern Munich acestea sunt decalate si pe axa posesiei, nu numai pe cea a suturilor spre poarta, deoarece am aproximat si posesia din meciul final, nu numai numarul de suturi. Consideram acest rezultat unul bun, observand distanta mica dintre cele doua puncte pe grafic.



2.1.4 Borussia Dortmund

Borussia Dortmund a jucat treisprezece meciuri in total in sezonul competitional al Ligii Campionilor 2012-2013, iar noi ne propunem sa aproximam finala: Borussia Dortmund - FC Bayern Munich 1 - 2, meci in care Borussia a avut o posesie de 42% si 12 suturi la poarta.

Graficul care reprezinta valoarea reala din finala, meci pe care dorim sa il aproximam, este urmatorul.



Suportul interpolarii

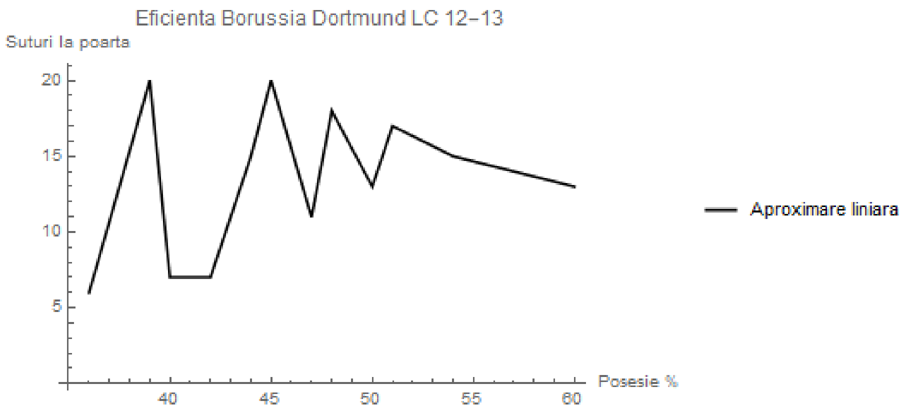
Suportul interpolarii va contine douasprezece puncte deoarece meciul din finala nu se ia in calcul, dorindu-ne sa il aproximam.

```
suportBorussia = {{36, 6}, {39, 20}, {40, 7}, {42, 7}, {44,
15}, {45, 20}, {47, 11}, {48, 18}, {50, 13}, {51, 17}, {54,
15}, {60, 13}}
```

Interpolare liniara

Incepem cu prima metoda de interpolare, cea liniara.

```
plotLiniarBorussia = ListPlot[
     suportBorussia,
     AxesOrigin->{35, 0},
     AxesLabel->{"Posesie %", "Suturi la poarta"},
     PlotLabel->"Eficienta Borussia Dortmund LC 12-13",
     Joined->True,
     PlotStyle->Black,
     PlotLegends->{"Aproximare liniara"}
]
```



Observam o forma care oscileaza, avand media suturilor spre poarta undeva intre 13 si 15, urcand deasupra si coborand sub acest interval la aproape fiecare doua puncte consecutive.

Interpolare Lagrange

Urmatoarea modalitate de interpolare este prin polinomul Lagrange care va avea cel mult grad 11, folosindu-ne de 12 puncte din suportul interpolarii.

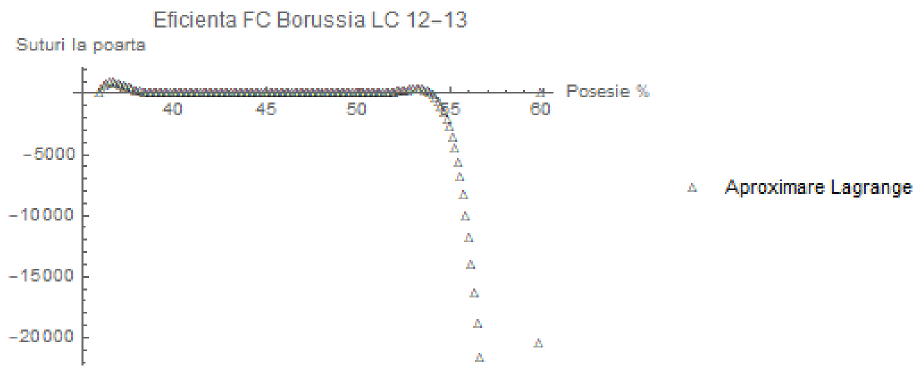
Codul de generare in Wolfram Mathematica a acestui polinom este urmatorul.

```
functieLagrangeBorussia = InterpolatingPolynomial[
     suportBorussia,
     x
]

puncteLagrangeBorussia = Table[
     functieLagrangeBorussia,
     {x, 36, 60, 0.15}
]
```

Generam puncte cu un pas de 0.15 pentru a putea observa bine graficul intre 36 si 60, valori cunoscute si de margine ale suportului interpolarii.

```
plotLagrangeBorussia = ListPlot[
     puncteLagrangeBorussia,
     DataRange -> {36, 60},
     AxesOrigin -> 35, 0,
     AxesLabel->{"Posesie %", "Suturi la poarta"},
     PlotLabel->"Eficienta Borussia Dortmund LC 12-13",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare Lagrange"},
     PlotMarkers -> {"\[EmptyUpTriangle]"}
]
```

Acest polinom oscileaza exagerat de mult in capatul din dreapta al intervalului folosit si asta inseamna ca daca posesia medie in care vom aproxima va fi foarte apropiata de acel capat, va trebui sa ignoram aceasta metoda.



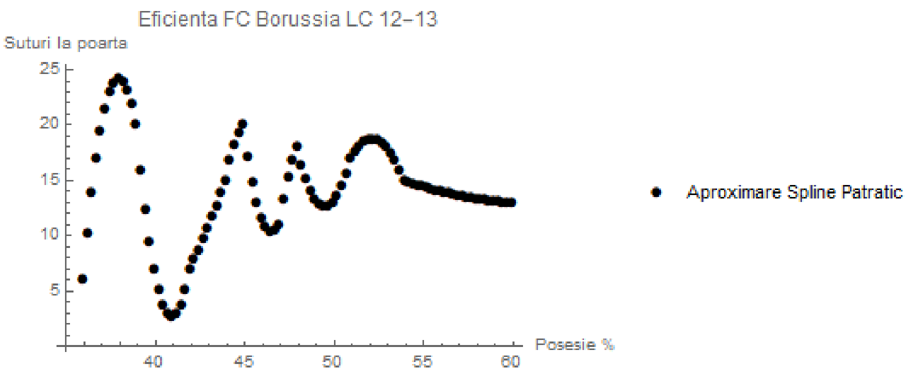
Interpolari de tip spline

Acum ne vom ocupa de abordarea cu functii spline, folosind codul Wolfram Mathematica urmator.

```
functieSplinePatraticBorussia = Interpolation[
     suportBorussia,
     InterpolationOrder->2][x]
]

puncteSplinePatraticBorussia = Table[
     functieSplinePatraticBorussia,
     {x, 36, 60, 0.25}
]

plotSplinePatraticBorussia = ListPlot[
     puncteSplinePatraticBorussia,
     DataRange -> {36, 60},
```

```
     AxesOrigin -> {35, 0},
     AxesLabel->{"Posesie %", "Suturi la poarta"},
     PlotLabel->"Eficienta Borussia Dortmund LC 12-13",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare Spline Patratic"},
     PlotMarkers -> {"\[FilledCircle]"}
]
```

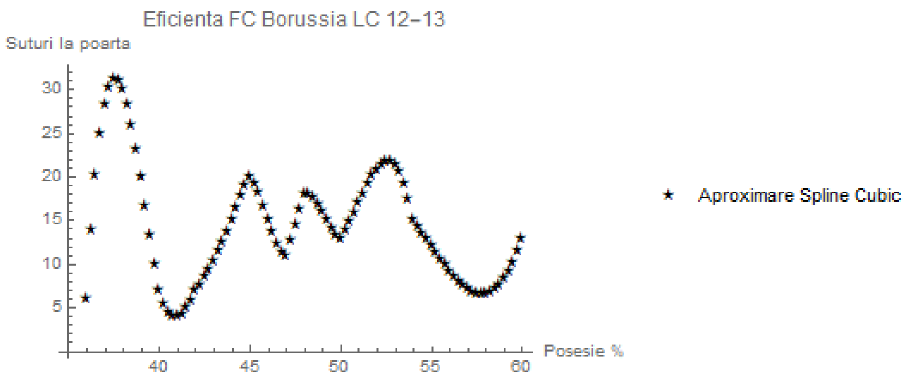
Dupa ce generam punctele care vor reprezenta interpolarea spline cu functii de imbinare patratice, le plotam pentru a avea o imagine mai ampla asupra statisticilor colectate.



In continuare, pregatim punctele pentru interpolarea spline care foloseste functii de imbinare cubice.

```
functieSplineCubicBorussia = Interpolation[
     suportBorussia,
     InterpolationOrder->3][x]

puncteSplineCubicBorussia = Table[
     functieSplineCubicBorussia,
     {x, 36, 60, 0.25}
]

plotSplineCubicBorussia = ListPlot[
```

```
    puncteSplineCubicBorussia,
    DataRange -> {36, 60},
    AxesOrigin -> {35, 0},
    AxesLabel->{"Posesie %", "Suturi la poarta"},
    PlotLabel->"Eficienta Borussia Dortmund LC 12-13",
    PlotStyle->Black,
    PlotLegends -> {"Aproximare Spline Cubic"},
    PlotMarkers -> {"\[FivePointedStar]"}
]
```



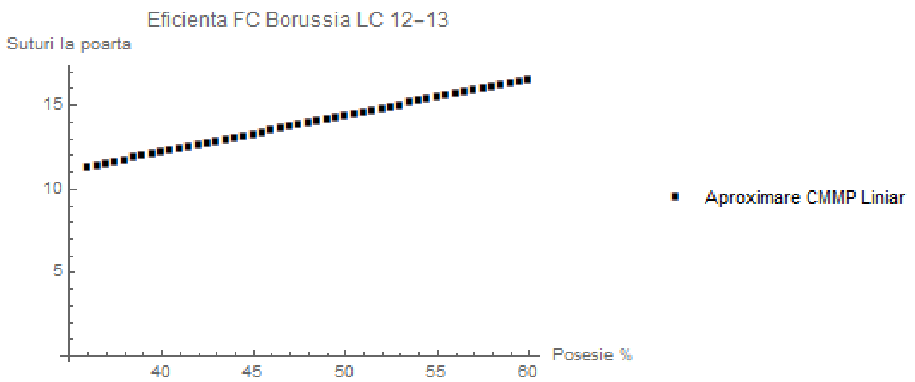
Observam ca spre deosebire de metoda care foloseste functii de grad doi, aceasta este mai permisiva in jurul punctelor care fac parte din suportul interpolarii si aproximeaza o plaja mai larga de valori pe axa suturilor spre poarta.

Aproximari in sensul CMMP

Ne ocupam in continuare de aproximarea in sensul celor mai mici patrate. Urmatorul cod Wolfram Mathematica pregateste punctele care alcatuiesc aproximarea liniara in sensul celor mai mici patrate. Uitandu-ne la graficul interpolarii liniare observam ca oscilatiile sunt mult mai mari la inceput si mai mici inspre final. Asta inseamna ca stim incotro se va indrepta aproximarea in sensul celor mai mici patrate, insa nu stim exact de unde va pleca.

```
functieCMMPLiniarBorussia = Fit[suportBorussia, {1, x}, x]

puncteCMMPLiniarBorussia = Table[
     functieCMMPLiniarBorussia,
     {x, 36, 60, 0.4}
]

plotCMMPLiniarBorussia = ListPlot[
     puncteCMMPLiniarBorussia,
     DataRange -> {36, 60},
     AxesOrigin -> {35, 0},
     AxesLabel->{"Posesie %", "Suturi la poarta"},
     PlotLabel->"Eficienta Borussia Dortmund LC 12-13",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare CMMP Liniar"},
     PlotMarkers -> {"\[FilledSquare]"}
]
```

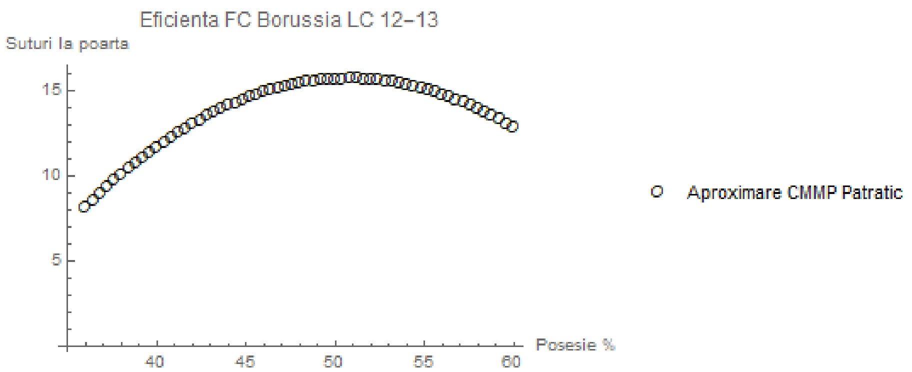
Graficul aproximarii liniare in sensul celor mai mici patrate arata ca mai jos.



Pentru aproximarea patratica in sensul celor mai mici patrate ne asteptam, mai degraba, la o curba bine definita deoarece am observat ca punctele sunt distribuite amortizat de la oscilatii mari la mici de la stanga la dreapta.

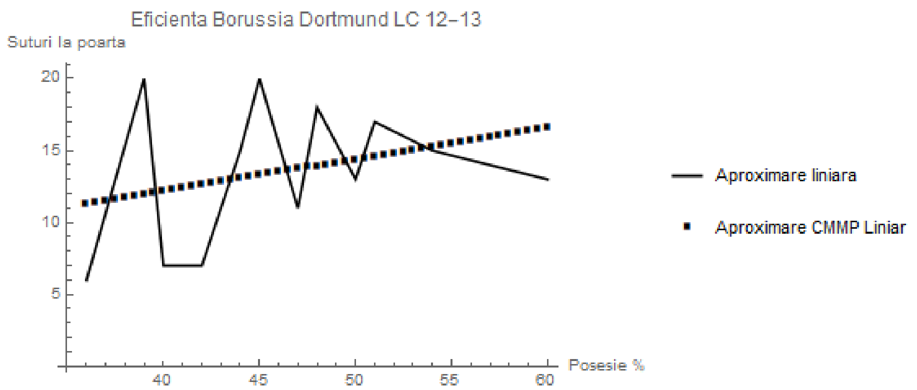
```
functieCMMPPatraticBorussia = Fit[suportBorussia, {1, x, xˆ2},
x]

puncteCMMPPatraticBorussia = Table[
     functieCMMPPatraticBorussia,
     {x, 36, 60, 0.5}
]

plotCMMPPatraticBorussia = ListPlot[
     puncteCMMPPatraticBorussia,
     DataRange -> {36, 60},
     AxesOrigin -> {35, 0},
     AxesLabel->{"Posesie %", "Suturi la poarta"},
     PlotLabel->"Eficienta Borussia Dortmund LC 12-13",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare CMMP Patratic"},
     PlotMarkers -> {"O"}
]
```

Graficul aproximarii patratice in sensul celor mai mici patrate este urmatorul si, dupa cum ne asteptam, este o curba cu o concavitate pronuntata.

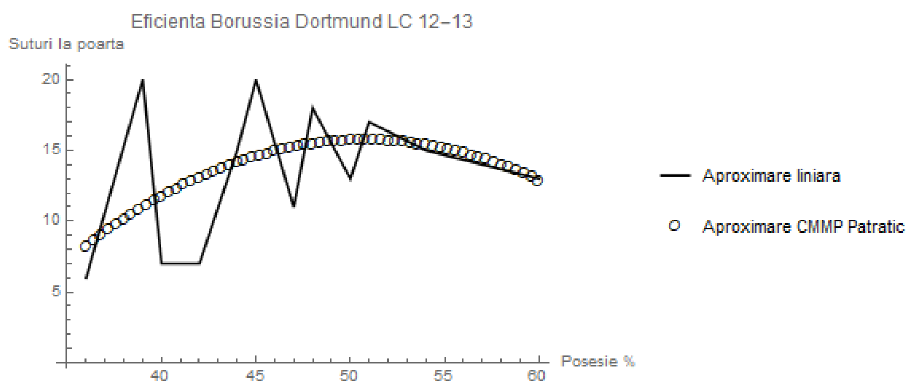


Comparam cele doua aproximari calculate in sensul celor mai mici patrate cu interpolarea liniara calculata anterior.

```
Show[plotLiniarBorussia, plotCMMPLiniarBorussia]
```

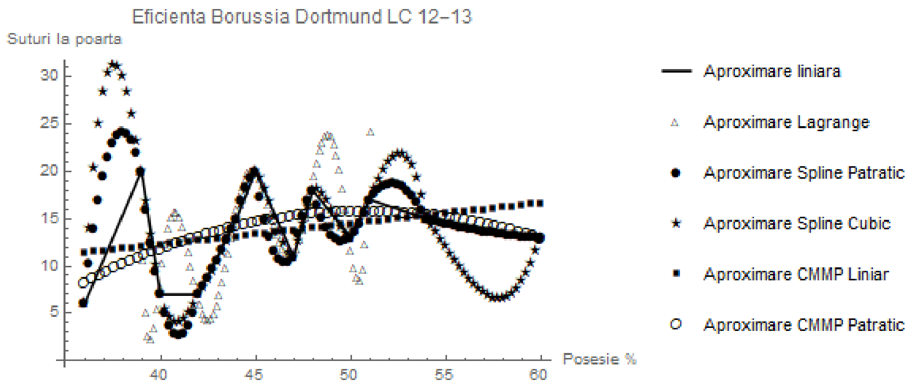


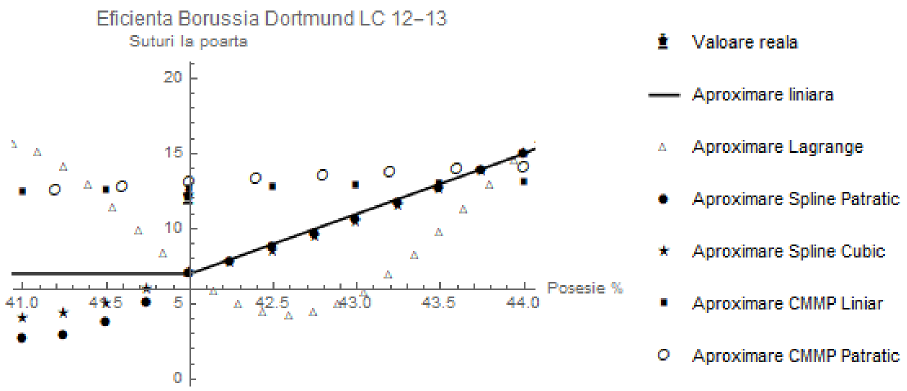

```
Show[plotLiniarBorussia, plotCMMPLiniarBorussia]
```



Dupa ce am construit toate interpolarile si aproximarile propuse urmeaza sa le afisam in acelasi grafic, iar apoi sa il centram in punctul reprezentand valoarea reala pe care o cautam ($\{42, 12\}$), adica posesia si numarul de suturi spre poarta din finala.

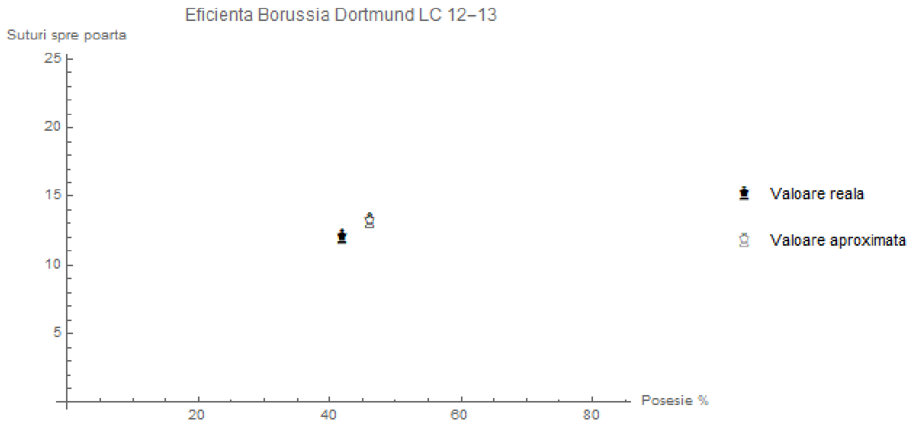
```
Show[
    plotLiniarBorussia,
    plotLagrBorussia,
    plotSplinePatraticBorussia,
    plotSplineCubicBorussia,
    plotCMMPLiniarBorussia,
    plotCMMPPatraticBorussia,
    PlotRange -> {{50, 60}, {0, 20}},
    AxesOrigin -> {42, 12}
]
```





Vom folosi toate rezulatele obtinute, astfel, in punctul de coordonata $x = 46.3$, adica media absciselor din suportul interpolarii, vom obtine $y = 13.1$.

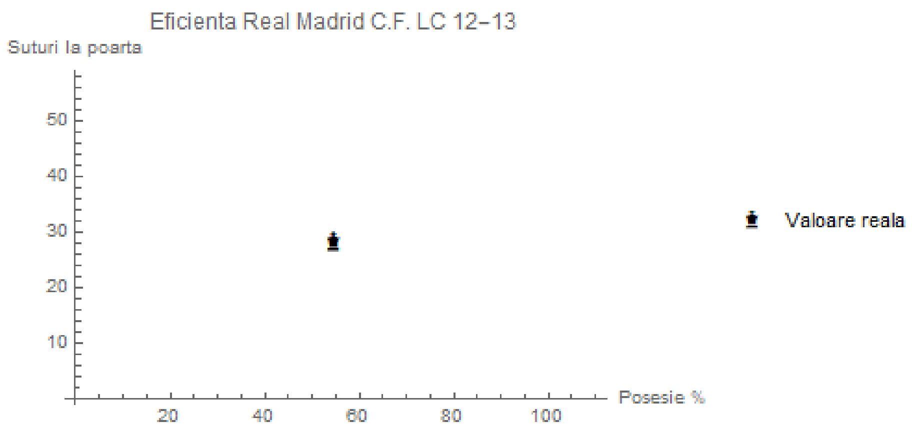
Observam ca distanta dintre valoarea reala si valoarea aproximata este egala cu 1, un rezultat fabulos datorat in mare parte pozitiei punctelor din suportul interpolarii.



2.1.5 Real Madrid C.F.

Ultima echipa ale carei rezultate dorim sa le aproximam in acest sezon competitional, 2012-2013, este Real Madrid C.F. Aceasta a jucat douasprezece meciuri in total, fiind eliminata in faza semifinalelor. Noi ne propunem sa aproximam numarul de suturi spre poarta din primul meci de dupa iesirea din grupe: Real Madrid C.F. - Manchester United FC 1 - 1. Real a reusit sa aiba o posesie de 55% si 28 de suturi spre poarta in acest meci.

Valoarea reala pe care dorim sa o aproximam se gaseste in graficul de mai jos.



Suportul interpolarii

Din suportul interpolarii vor face parte unsprezece puncte ordonate dupa posesie, adica primul termen.

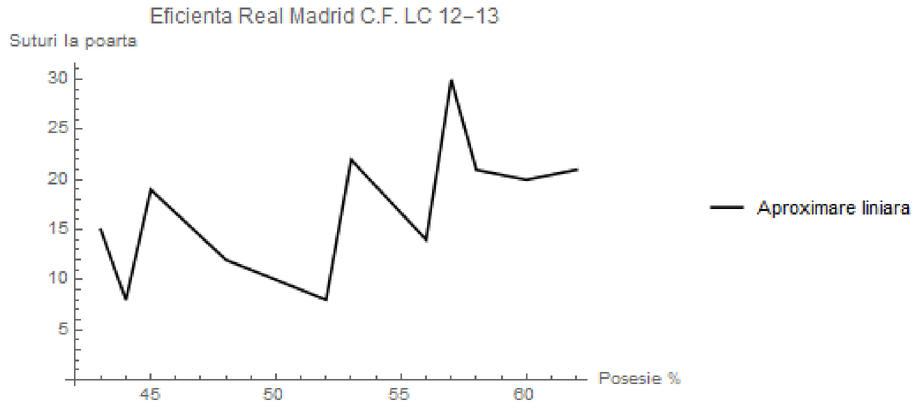
`suportReal = {{43, 15}, {44, 8}, {45, 19}, {48, 12}, {52, 8}, {53, 22}, {56, 14}, {57, 30}, {58, 21}, {60, 20}, {62, 21}}`

Interpolare liniara

Dorim sa realizam interpolarea liniara a punctelor din suport, iar acest lucru se poate obtine in Wolfram Mathematica astfel.

```
plotLiniarReal = ListPlot[
     suportReal,
     AxesOrigin->{42, 0},
     AxesLabel->{"Posesie %", "Suturi la poarta"},
     PlotLabel->"Eficienta Real Madrid C.F. LC 12-13",
     Joined->True,
     PlotStyle->Black,
     PlotLegends->{"Aproximare liniara"}
]
```

Urmatoarea figura reprezinta graficul interpolarii liniare a punctelor ce apartin suportului interpolarii.



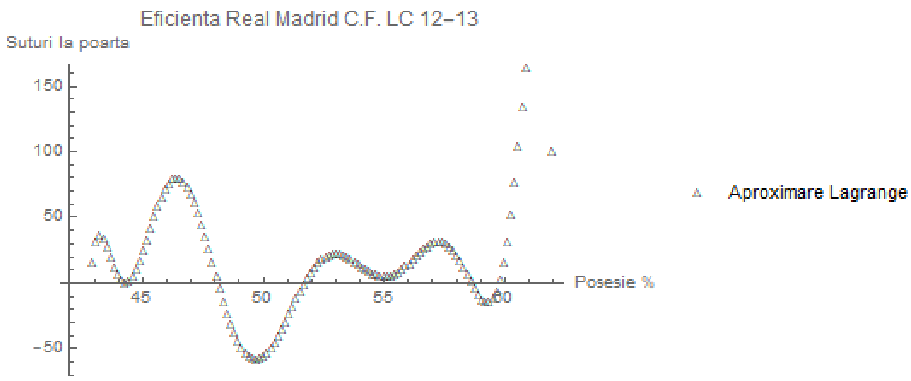
Interpolare Lagrange

Acum ne propunem sa construim polinomul Lagrange de interpolare care va avea grad maxim 10, deoarece este generat folosind 11 puncte. Acesta va fi construit intre punctele 43 si 62, adica cea mai mica si cea mai mare posesie pe care le-a obtinut Real Madrid C.F. intr-un meci din acest sezon competitional.

```
functieLagrangeReal = InterpolatingPolynomial[
     suportReal,
     x
]

puncteLagrangeReal = Table[
     functieLagrangeReal,
     {x, 43, 62, 0.15}
]

plotLagrangeReal = ListPlot[
     puncteLagrangeReal,
     DataRange -> {43, 62},
```

```
    AxesOrigin -> 42, 0,
    AxesLabel->{"Posesie %", "Suturi la poarta"},
    PlotLabel->"Eficienta Real Madrid C.F. LC 12-13",
    PlotStyle->Black,
    PlotLegends -> {"Aproximare Lagrange"},
    PlotMarkers -> {"\[EmptyUpTriangle]"}
]
```

Graficul reprezentativ pentru interpolarea Lagrange este urmatorul. Se poate observa ca pentru aceasta echipa oscilatiile polinomului sunt mult mai mici, desi in capete tot sunt nerealiste, 150 de suturi spre poarta fiind o valoare aproape imposibila intr-un meci real.



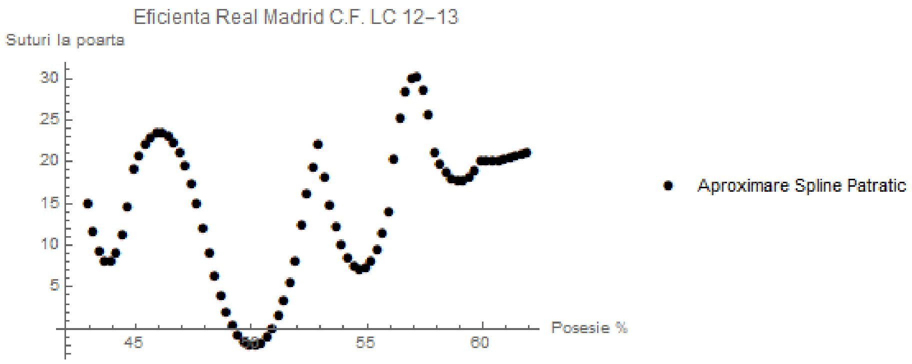
Interpolari de tip spline

Construim cele doua interpolari spline, cu functii de imbinare patratice si cubice.

```
functieSplinePatraticReal = Interpolation[      suportReal,
    InterpolationOrder->2][x]
]

puncteSplinePatraticReal = Table[
    functieSplinePatraticReal,
```

```
    {x, 43, 62, 0.25}
]

plotSplinePatraticReal = ListPlot[
    puncteSplinePatraticReal,
    DataRange -> {43, 62},
    AxesOrigin -> {42, 0},
    AxesLabel->{"Posesie %", "Suturi la poarta"},
    PlotLabel->"Eficienta Real Madrid C.F. LC 12-13",
    PlotStyle->Black,
    PlotLegends -> {"Aproximare Spline Patratic"},
    PlotMarkers -> {"\[FilledCircle]"}
]
```

Observam in graficul pentru interpolare spline cu functii de imbinare patratice ca, spre deosebire de celelalte echipe discutate pana acum, acesta obtine valori negative pe o anumita portiune, lucru imposibil in viata reala. Astfel, ne confirmam faptul ca oricat de mult ar reduce oscilatiile produse de interpolari precum Lagrange, tot este posibil sa obtii rezultate nerealiste.



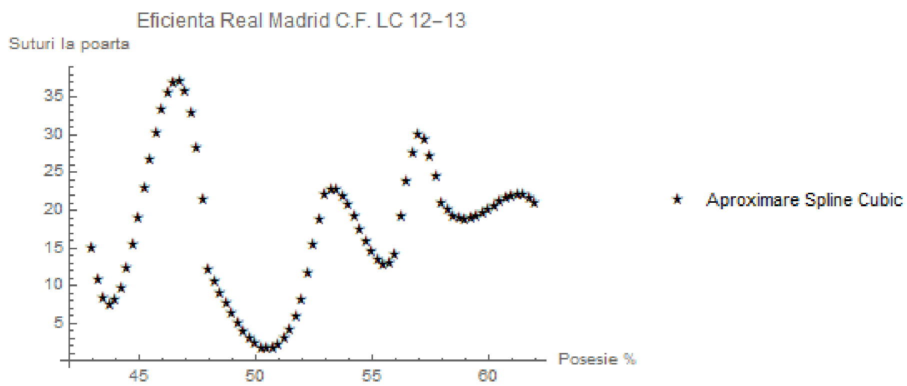
Construim si interpolarea spline cu functii de imbinare cubice si asteptam rezultate asemanatoare cu abordarea anterioara.

```
functieSplineCubicReal = Interpolation[
     suportReal,
     InterpolationOrder->3][x]

puncteSplineCubicReal = Table[
     functieSplineCubicReal,
     {x, 43, 62, 0.25}
]
```

Graficul este plotat cu originea in punctul {42, 0} pentru a putea observa toate punctele din suportul interpolarii.

```
plotSplineCubicReal = ListPlot[
     puncteSplineCubicReal,
     DataRange -> {43, 62},
     AxesOrigin -> {42, 0},
     AxesLabel->{"Posesie %", "Suturi la poarta"},
     PlotLabel->"Eficienta Real Madrid C.F. LC 12-13",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare Spline Cubic"},
     PlotMarkers -> {"\[FivePointedStar]"}
]
```

Graficul acestei metode este urmatorul.

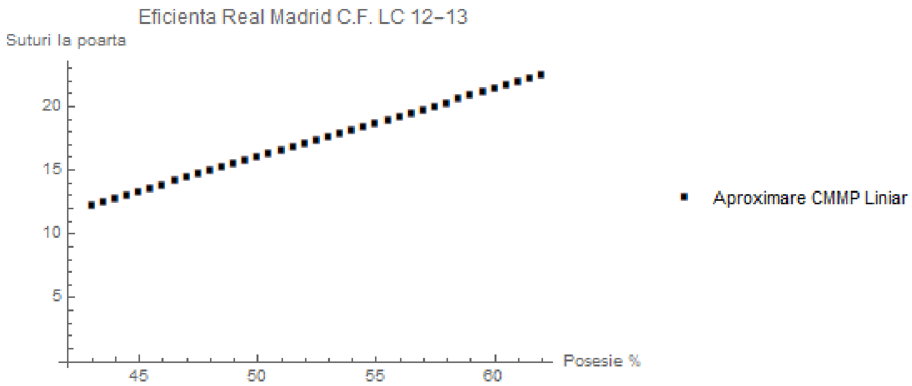


Aproximari in sensul CMMP

Pentru a aproxima cu un polinom liniar in sensul celor mai mici patrate e nevoie sa gasim un polinom pentru care suma distantelor de la punctele din suportul interpolarii la el este minima.

```
functieCMMPLiniarReal = Fit[suportReal, {1, x}, x]
puncteCMMPLiniarReal = Table[
     functieCMMPLiniarReal,
     {x, 43, 62, 0.4}
]

plotCMMPLiniarReal = ListPlot[
     puncteCMMPLiniarReal,
     DataRange -> {43, 62},
     AxesOrigin -> {42, 0},
     AxesLabel->{"Posesie %", "Suturi la poarta"},
     PlotLabel->"Eficienta Real Madrid C.F. LC 12-13",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare CMMP Liniar"},
     PlotMarkers -> {"\[FilledSquare]"}
]
```

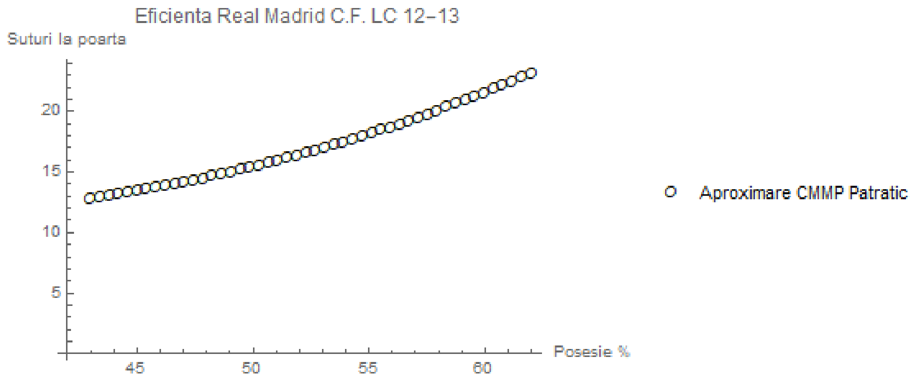
Graficul aproximarii liniare este ca cel de mai jos.



Aproximarea printr-un polinom patratic ales in sensul celor mai mici patrate care sa treaca printre toate punctele din suportul interpolarii se realizeaza in Wolfram Mathematica astfel.

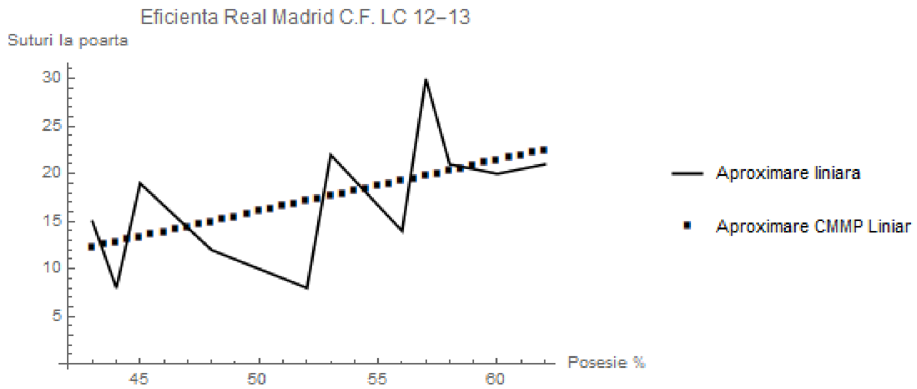
```
functieCMMPPatraticReal = Fit[suportReal, {1, x, x^2}, x]
puncteCMMPPatraticReal = Table[
     functieCMMPPatraticReal,
     {x, 43, 62, 0.5}
]

plotCMMPPatraticReal = ListPlot[
     puncteCMMPPatraticReal,
     DataRange -> {43, 62},
     AxesOrigin -> {42, 0},
     AxesLabel->{"Posesie %", "Suturi la poarta"},
     PlotLabel->"Eficienta Real Madrid C.F. LC 12-13",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare CMMP Patratic"},
     PlotMarkers -> {"O"}
]
```

Graficul aproximarii patratice este cel de mai jos.

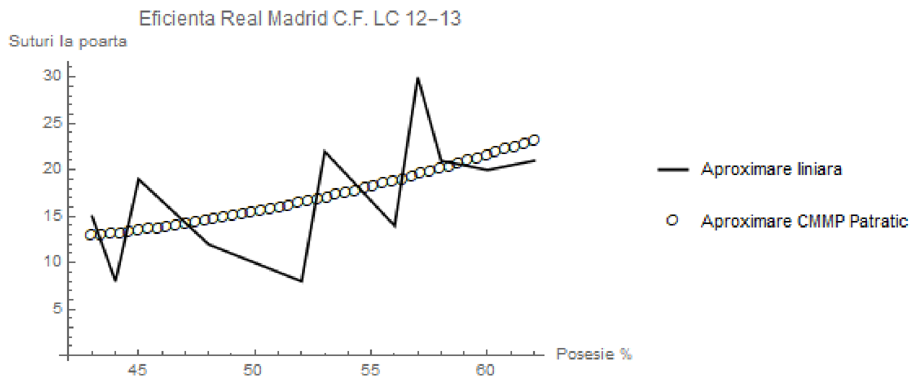


Urmeaza sa suprapunem graficul interpolarii liniare cu graficele aproximarilor in sensul celor mai mici patrate.

```
Show[plotLiniarReal, plotCMMPLiniarReal]
```

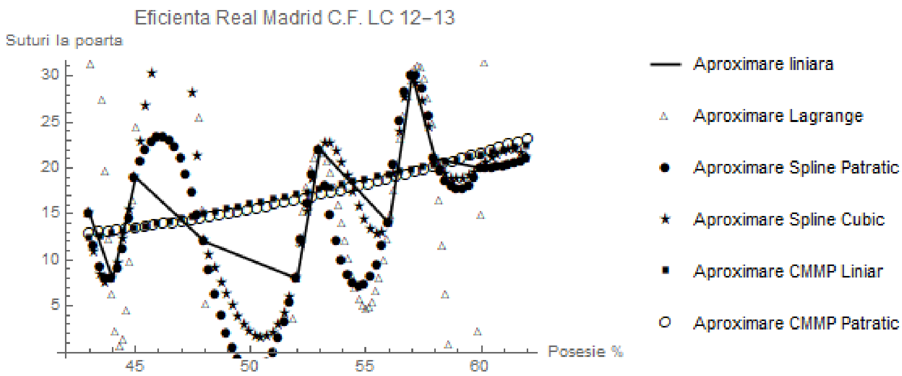


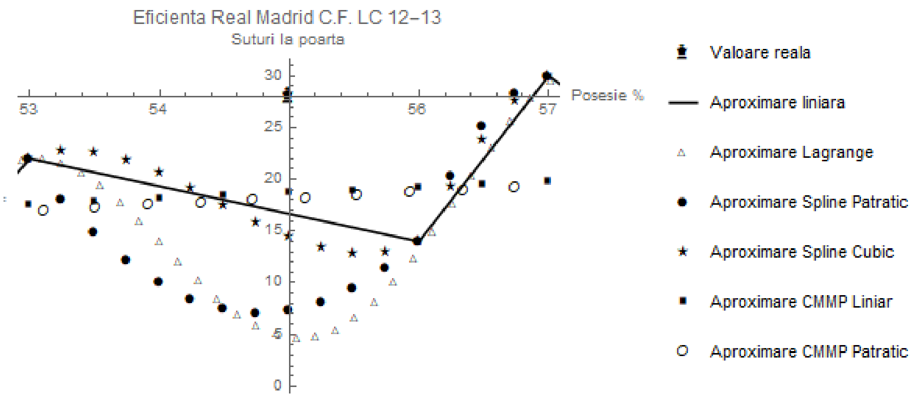

```
Show[plotLiniarReal, plotCMMPLiniarReal]
```



Dupa ce am construit toate interpolarile si aproximarile propuse urmeaza sa le afisam in acelasi grafic, iar apoi sa il centram in punctul reprezentand valoarea reala pe care o cautam ($\{55, 28\}$), adica posesia si numarul de suturi la poarta din primul meci de dupa faza grupelor. In Wolfram Mathematica acest lucru se va face in modul urmator.

```
Show[
    plotLiniarReal,
    plotLagrReal,
    plotSplinePatraticReal,
    plotSplineCubicReal,
    plotCMMPLiniarReal,
    plotCMMPPatraticReal,
    PlotRange -> {{53, 57}, {0, 30}},
    AxesOrigin -> {55, 28}
]
```

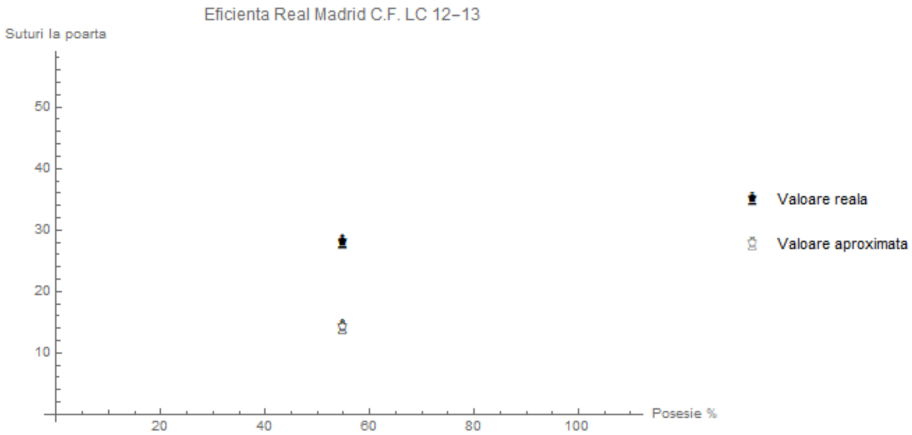




Vom folosi toate rezulatele obtinute, astfel, in punctul de coordonata $x = 55$, adica valoarea reala a posesiei din meciul urmarit, vom obtine

$y = 14$. Aceasta valoare este destul de indepartata de valoarea reala si urmarind graficul anterior observam ca toate metodele sunt destul de departe de ce ar fi trebuit sa aproximeze. Acest rezultat ne reaminteste ca fotbalul este un joc din care nu lipseste hazardul si trebuie sa fim de acord si cu astfel de exceptii.

Valoarea reala si valoarea aproximata puse una langa alta arata ca mai jos.



2.2 Liga Campionilor 2013-2014

In acest an ne-am propus sa urmarim evolutia suturilor la poarta fata de numarul de pase complete.

Am selectat urmatoarele echipe:

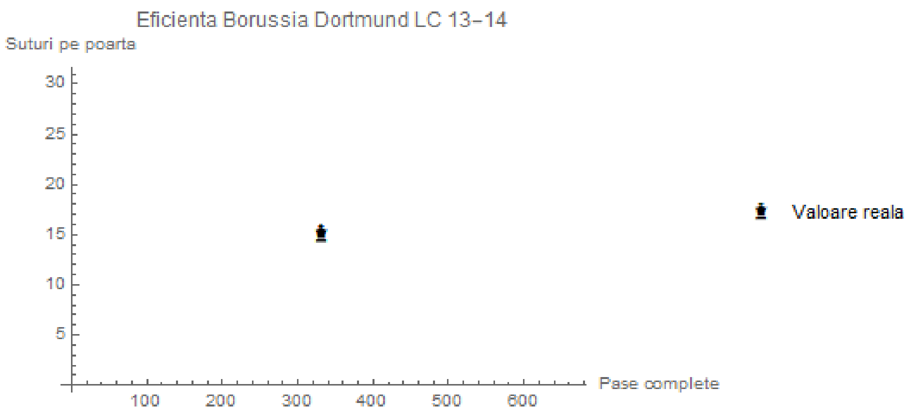
- Borussia Dortmund
- FC Barcelona
- FC Bayern Munich
- Paris Saint-Germain F.C.
- Real Madrid C.F.

Deoarece ordinul de marime la care vor fi raportate abscisele punctelor din suportul interpolarii este mai mare decat in subcapitolul trecut, unde urmaream procentajul posesiei, insa numarul total de puncte cunoscute ramane aproximativ la fel, vom incerca sa observam cum se comporta metodele de interpolare si aproximare si daca acestea oscileaza mai mult sau din contra.

2.2.1 Borussia Dortmund

Datorita faptului ca Borussia Dortmund a jucat 10 meciuri in sezonul competitional 2013-2014 al Ligii Campionilor, fiind eliminata in sferturile competitiei de Real Madrid, echipa care urma sa ajunga campioana din acel an, ne dorim sa aproximam numarul de suturi spre poarta din primul meci de dupa faza grupelor, FC Zenit Saint Petersburg - Borussia Dortmund 2 - 4. In acest meci, nemtii au reusit 334 de pase complete si 15 suturi.

Graficul care evidentiaza valoarea reala pe care dorim sa o aproximam este urmatorul.



Suportul interpolarii

Avem noua puncte in suportul interpolarii, incercand sa aproximam al zecelea rezultat cunoscut. De data aceasta abscisele punctelor sunt reprezentate de sute in loc de zeci.

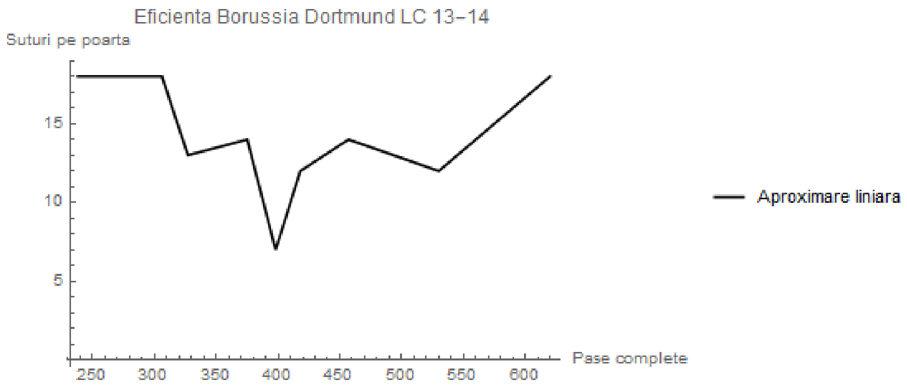
```
suportBorussia = {{238, 18}, {306, 18}, {327, 13}, {375, 14},
{398, 7}, {418, 12}, {457, 14}, {530, 12}, {620, 18}}
```

Interpolare liniara

Incepem cu interpolarea liniara si observam ca procedeul de generare a acesteia in Wolfram Mathematica este similar cu cel din subcapitolul trecut.

```
plotLiniarBorussia = ListPlot[
     suportBorussia,
     AxesOrigin->{237, 0},
     AxesLabel->{"Pase complete", "Suturi pe poarta"},
     PlotLabel->"Eficienta Borussia Dortmund LC 13-14",
     Joined->True,
     PlotStyle->Black,
     PlotLegends->{"Aproximare liniara"}
]
```

Graficul acestei interpolari ne duce cu gandul la un polinom convex, coborand pana la mijlocul suportului de interpolare si revenind apoi, pana acum observand doar forme concave sau cu mai multe inflexiuni.



Interpolare Lagrange

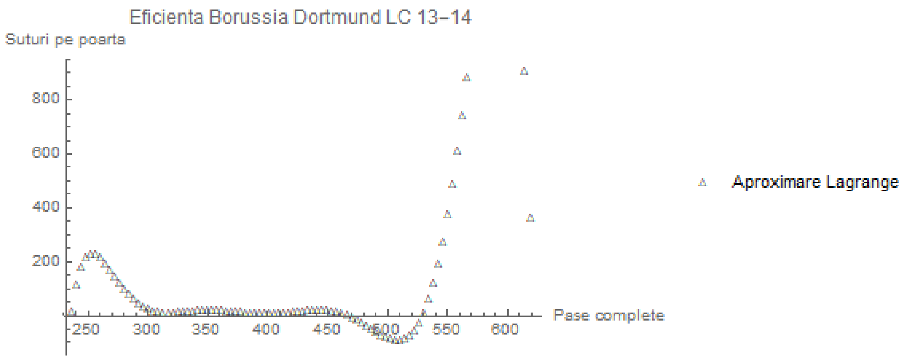
Pentru interpolarea Lagrange generam punctele pe care le vom afisa pe grafic si ne asteptam la oscilatii mari in capete deoarece distanta dintre doua puncte consecutive acum este de ordinul zecilor, nu al unitatilor.

```
functieLagrangeBorussia = InterpolatingPolynomial[
     suportBorussia,
     x
]
```

Setam ca margini 238 si 620, adica cel mai mic numar de pase complete efectuate intr-un meci si, respectiv, cel mai mare. Distanta 4 dintre puncte este pentru o mai buna vizualizare a graficului, fiindca daca am fi ales ceva mai mic s-ar fi aglomerat foarte multe puncte pe grafic.

```
puncteLagrangeBorussia = Table[
     functieLagrangeBorussia,
     {x, 238, 620, 4}
]

plotLagrangeBorussia = ListPlot[
     puncteLagrangeBorussia,
     DataRange -> {238, 620},
     AxesOrigin -> 237, 0,
     AxesLabel->{"Pase complete", "Suturi pe poarta"},
     PlotLabel->"Eficienta Borussia Dortmund LC 13-14",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare Lagrange"},
     PlotMarkers -> {"\[EmptyUpTriangle]"}
]
```

Dupa cum ne asteptam, graficul are oscilatii mari in capete, mai ales in capatul din dreapta, cum se poate observa si in graficul de mai jos.



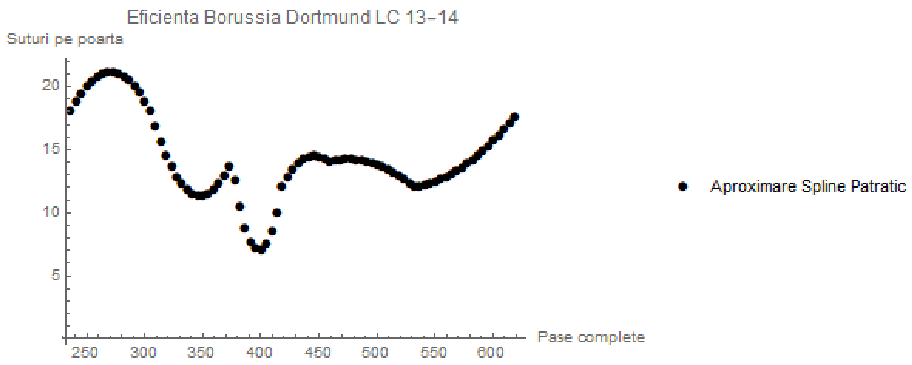
Interpolari de tip spline

Urmeaza interpolarile de tip spline care folosesc functii de imbinare patratice si cubice. Pentru generarea acestora in Wolfram Mathematica se efectueaza urmatoarele operatii.

```
functieSplinePatraticBorussia = Interpolation[
     suportBorussia,
     InterpolationOrder->2][x]
]

puncteSplinePatraticBorussia = Table[
     functieSplinePatraticBorussia,
     {x, 238, 620, 4.5}
]

plotSplinePatraticBorussia = ListPlot[
     puncteSplinePatraticBorussia,
     DataRange -> {238, 620},
     AxesOrigin -> {237, 0},
     AxesLabel->{"Pase complete", "Suturi pe poarta"},
     PlotLabel->"Eficienta Borussia Dortmund LC 13-14",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare Spline Patratic"},
```

```
    PlotMarkers -> {"\[FilledCircle]"}
]
```

Graficul interpolarii spline realizate cu ajutorul functiilor patratice de imbinare arata astfel.

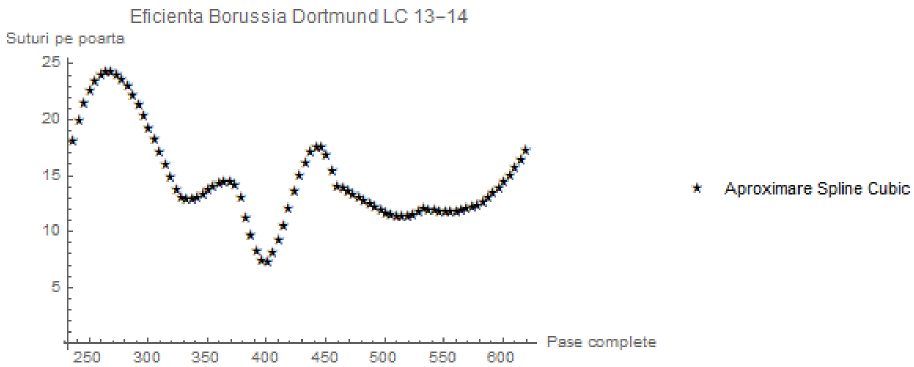


Ne ocupam in continuare de generarea interpolarii spline cu ajutorul functiilor cubice de imbinare.

```
functieSplineCubicBorussia = Interpolation[
    suportBorussia,
    InterpolationOrder->3][x]

puncteSplineCubicBorussia = Table[
    functieSplineCubicBorussia,
    {x, 238, 620, 4.5}
]

plotSplineCubicBorussia = ListPlot[
    puncteSplineCubicBorussia,
    DataRange -> {238, 620},
    AxesOrigin -> {237, 0},
    AxesLabel->{"Pase complete", "Suturi pe poarta"},
    PlotLabel->"Eficienta Borussia Dortmund LC 13-14",
```

```
    PlotStyle->Black,
    PlotLegends -> {"Aproximare Spline Cubic"},
    PlotMarkers -> {"\[FivePointedStar]"}
]
```

Graficul reprezentativ interpolarii cu functii cubice de imbinare este urmatorul.



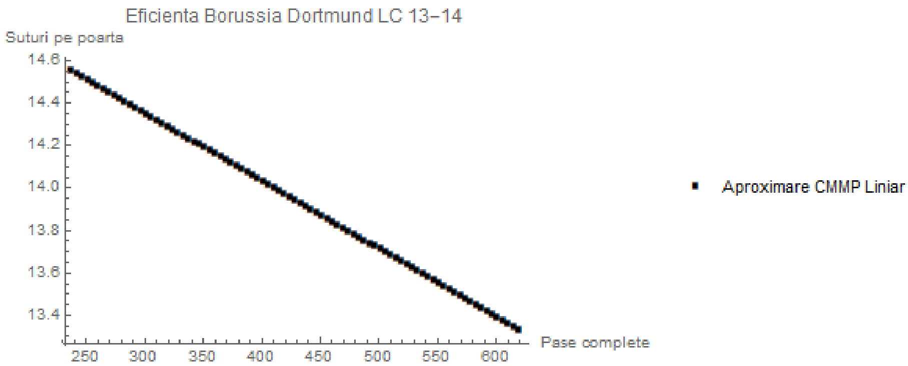
Aproximari in sensul CMMP

Dorim sa gasim o succesiune de puncte care trec printre toate punctele suportului interpolarii. Pentru inceput, vom incerca sa gasim o dreapta.

```
functieCMMPLiniarBorussia = Fit[suportBorussia, {1, x}, x]

puncteCMMPLiniarBorussia = Table[
    functieCMMPLiniarBorussia,
    {x, 238, 620, 4.5}
]

plotCMMPLiniarBorussia = ListPlot[
    puncteCMMPLiniarBorussia,
    DataRange -> {238, 620},
    AxesOrigin -> {237, 0},
```

```
    AxesLabel->{"Pase complete", "Suturi pe poarta"},
    PlotLabel->"Eficienta Borussia Dortmund LC 13-14",
    PlotStyle->Black,
    PlotLegends -> {"Aproximare CMMP Liniar"},
    PlotMarkers -> {"\[FilledSquare]"}
]
```

Aproximarea liniara in sensul celor mai mici patrate arata ca mai jos. Observam ca aceasta coboara, insa foarte putin desi graficul pare abrupt.



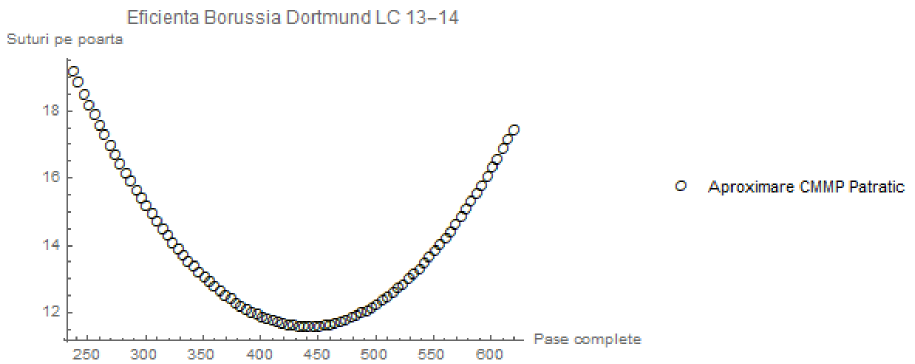
Cautam si un polinom patratic care sa treaca printre toate punctele suportului interpolarii.

```
functieCMMPPatraticBorussia = Fit[suportBorussia, {1, x, x^2},
x]

puncteCMMPPatraticBorussia = Table[
    functieCMMPPatraticBorussia,
    {x, 238, 620, 4.5}
]

plotCMMPPatraticBorussia = ListPlot[
    puncteCMMPPatraticBorussia,
    DataRange -> {238, 620},
    AxesOrigin -> {237, 0},
```

```
    AxesLabel->{"Pase complete", "Suturi pe poarta"},
    PlotLabel->"Eficienta Borussia Dortmund LC 13-14",
    PlotStyle->Black,
    PlotLegends -> {"Aproximare CMMP Patratic"},
    PlotMarkers -> {"O"}
]
```

Graficul aproximarii patratice in sensul celor mai mici patrate este urmatorul si putem observa ca este convex, exact ce banuiam la inceput.

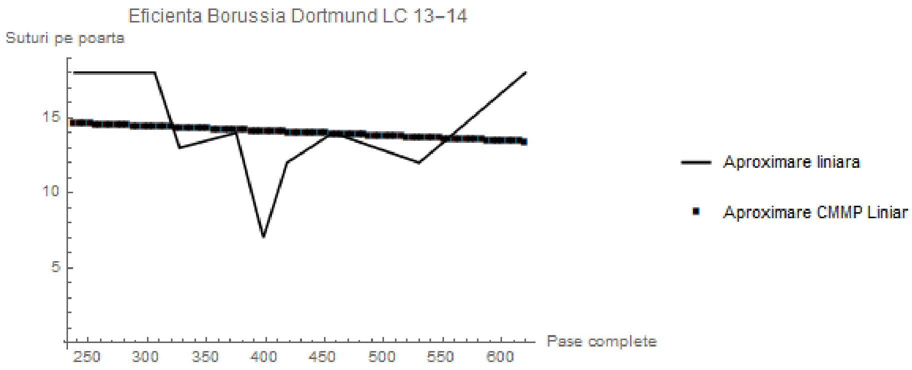
Acest lucru ne poate spune ca echipa este mai agresiva atunci cand este in una din cele doua extreme, cu foarte multe sau foarte putine pase complete.



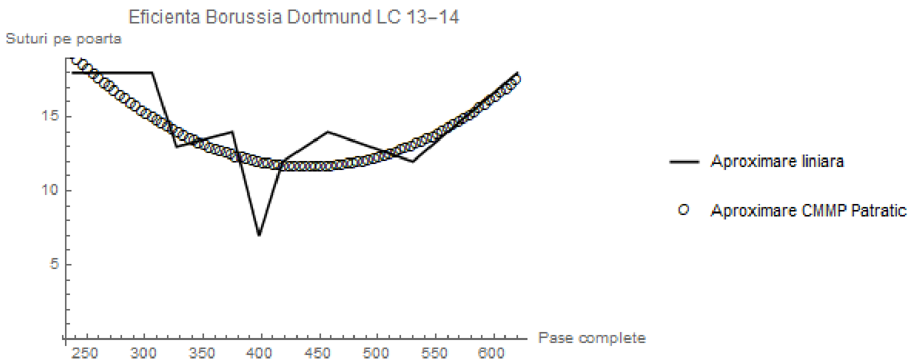
Urmeaza sa suprapunem graficele interpolarii liniare si aproximarilor in sensul celor mai mici patrate pentru a vedea la o scara mai mare daca se valideaza proprietatea de a trece printre toate punctele.

Avand deja plotarile efectuate ne mai ramane doar sa le afisam impreuna folosind functia "Show" din Wolfram Mathematica.

```
Show[
    plotLiniarBorussia,
    plotCMMPLiniarBorussia
]
```



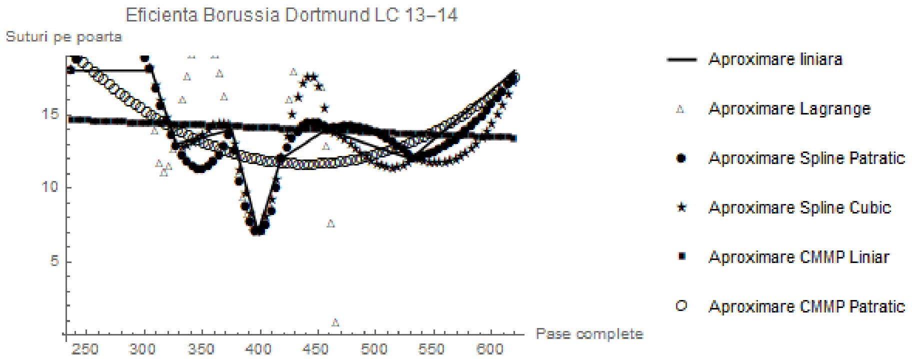

```
Show[
     plotLiniarBorussia,
     plotCMMPLiniarBorussia
]
```



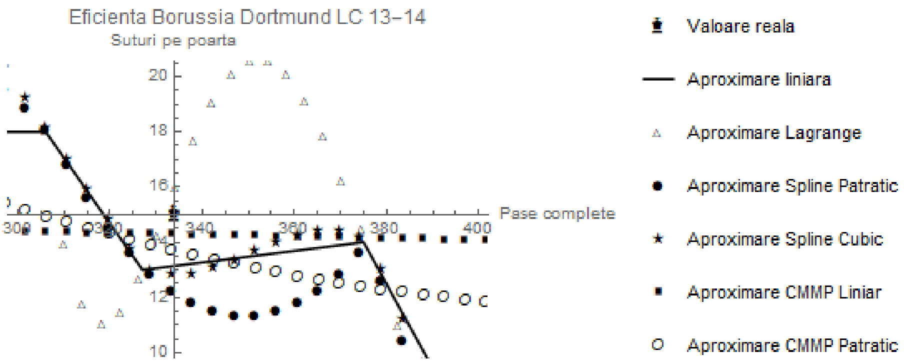
In continuare afisam toate aproximarile calculate si apoi centram in punctul {334, 15} care reprezinta valoarea reala pe care o cautam.

```
Show[
     plotLiniarBorussia,
     plotLagrBorussia,
     plotSplinePatraticBorussia,
     plotSplineCubicBorussia,
```

```
    plotCMMPLiniarBorussia,
    plotCMMPPatraticBorussia,
    PlotRange -> {{300, 400}, {10, 20}}
    AxesOrigin -> {334, 15}
]
```

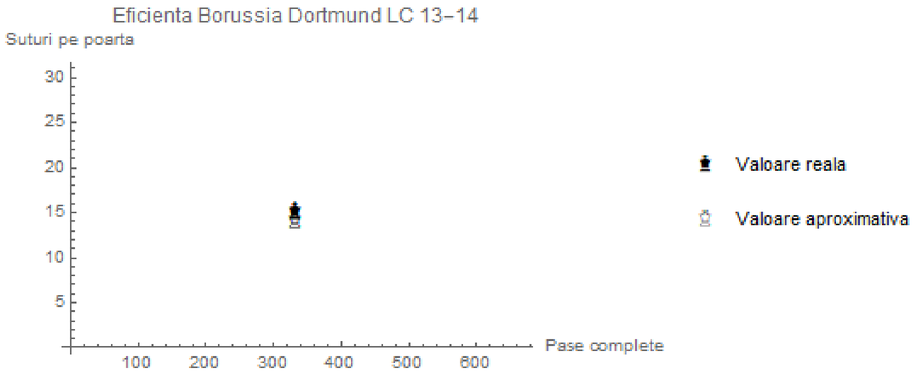


Observam in graficul centrat de mai jos ca toate aproximarile pot fi folosite pentru estimarea dorita deoarece sunt foarte apropiate de valoarea reala.



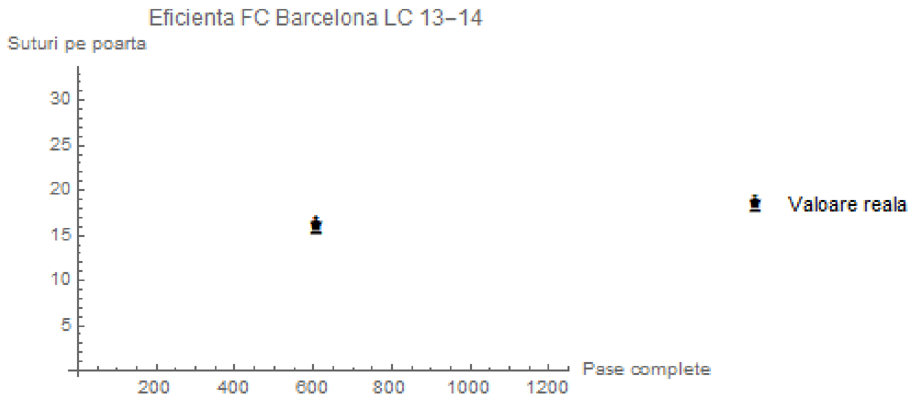
Gasim astfel o aproximare in punctul $\{334, 14\}$ care este la diferenta 1 de valoarea reala. Consideram acest rezultat unul uimitor care ne confirma ca ordinul de marime al punctelor din suportul interpolarii nu este neaparat un factor decisiv in estimarea unei valori.

Valoarea reala si valoarea aproximativa arata grafic astfel.



2.2.2 FC Barcelona

Dorim sa aproximam numarul de suturi spre poarta ale echipei FC Barcelona in al doilea meci de dupa faza grupelor in sezonul competitional 2013-2014: FC Barcelona - Manchester City F.C. 2 - 1. Spaniolii au jucat in total zece meciuri in aceasta editie a Ligii Campionilor, fiind eliminati in sferturi de Atletico Madrid. Valoarea reala a statisticilor dorite (pase complete/suturi spre poarta) in meciul urmarit este {612, 16}.



Suportul interpolarii

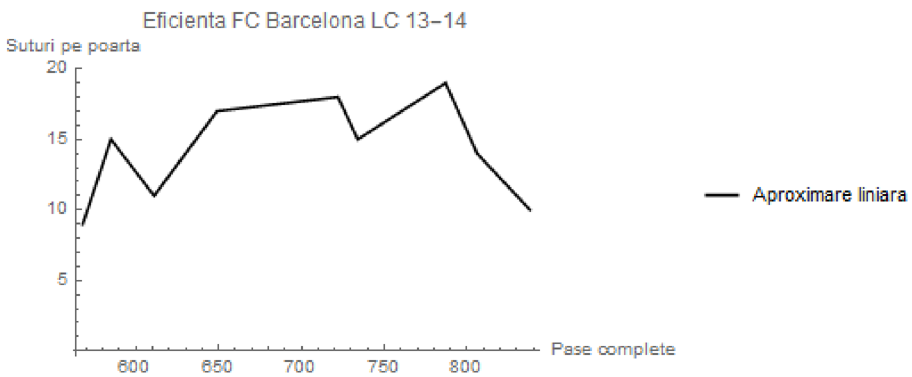
Suportul interpolarii este format din statisticile celorlalte noua meciuri pe care FC Barcelona le-a jucat in sezonul 2013-2014.

```
suportBarcelona = {{568, 9}, {585, 15}, {611, 11}, {649, 17},
{722, 18}, {734, 15}, {787, 19}, {806, 14}, {838, 10}}
```

Interpolare liniara

Incepem cu interpolarea liniara care ne arata o imagine de ansamblu a progresului statisticilor urmarite de-a lungul meciurilor jucate.

```
plotLiniarBarcelona = ListPlot[
     suportBarcelona,
     AxesOrigin->{567, 0},
     AxesLabel->{"Pase complete", "Suturi pe poarta"},
     PlotLabel->"Eficienta FC Barcelona LC 13-14",
     Joined->True,
     PlotStyle->Black,
     PlotLegends->{"Aproximare liniara"}
]
```



Interpolare Lagrange

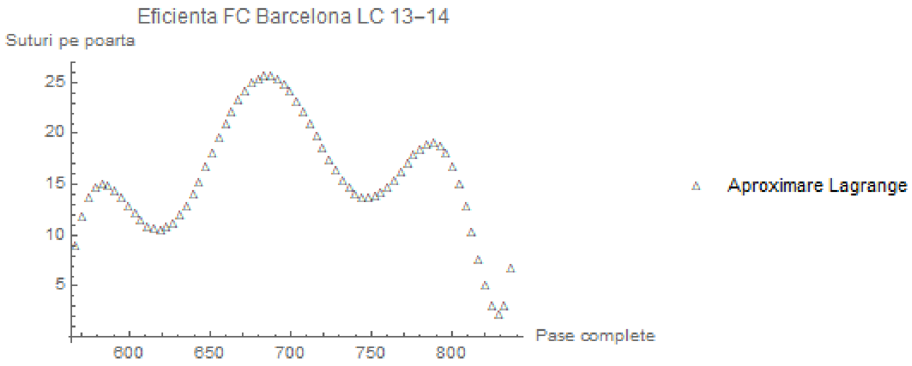
Continuam cu interpolarea Lagrange prin punctele din suportul interpolarii.

Observand diferentele mici de valoare pe axa suturilor spre poarta intre oricare doua puncte consecutive folosind interpolarea liniara, ne putem astepta, cu toate ca nu este sigur, la oscilatii mai mici din partea polinomului de interpolare Lagrange.

Codul Wolfram Mathematica care genereaza acest polinom de interpolare de grad maxim opt, deoarece sunt noua puncte in suport, este urmatorul.

```
functieLagrangeBarcelona = InterpolatingPolynomial[
     suportBarcelona,
     x
]

puncteLagrangeBarcelona = Table[
     functieLagrangeBarcelona,
     {x, 568, 838, 4}
]

plotLagrangeBarcelona = ListPlot[
     puncteLagrangeBarcelona,
     DataRange -> {568, 838},
     AxesOrigin -> 567, 0,
     AxesLabel->{"Pase complete", "Suturi pe poarta"},
     PlotLabel->"Eficienta FC Barcelona LC 13-14",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare Lagrange"},
     PlotMarkers -> {"\[EmptyUpTriangle]"}
]
```

Graficul interpolarii Lagrange arata ca in figura urmatoare si observam ca presupunerea de mai devreme a fost corecta, astfel incat oscilatiile produse sunt mici, chiar si la capetele suportului de puncte, lucru atipic interpolarii de acest tip.



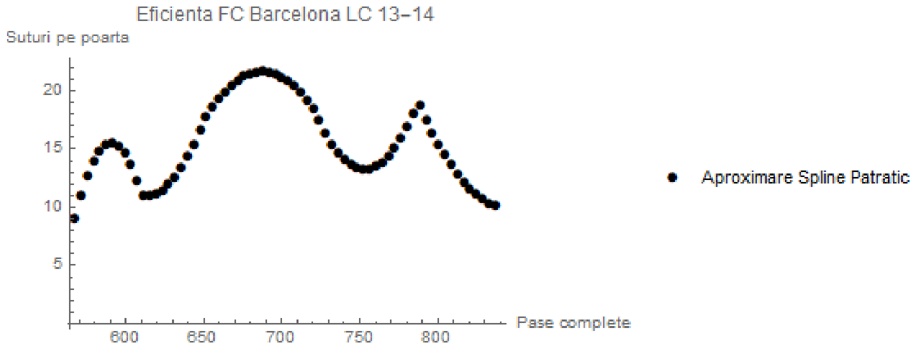
Interpolari de tip spline

Acum vom genera interpolarile de tip spline cu ajutorul functiei "Interpolation" din Wolfram Mathematica, mentionand ordinul functiilor de imbinare, 2 sau 3, in functie de caz.

```
functieSplinePatraticBarcelona = Interpolation[
     suportBarcelona,
     InterpolationOrder->2][x]
]

puncteSplinePatraticBarcelona = Table[
     functieSplinePatraticBarcelona,
     {x, 568, 838, 4}
]

plotSplinePatraticBarcelona = ListPlot[
     puncteSplinePatraticBarcelona,
     DataRange -> {568, 838},
     AxesOrigin -> {567, 0},
     AxesLabel->{"Pase complete", "Suturi pe poarta"},
```

```
    PlotLabel->"Eficienta FC Barcelona LC 13-14",
    PlotStyle->Black,
    PlotLegends -> {"Aproximare Spline Patratic"},
    PlotMarkers -> {"\[FilledCircle]"}
]
```

Graficul interpolarii spline cu ajutorul functiilor patratice de imbinare arata in felul urmator.

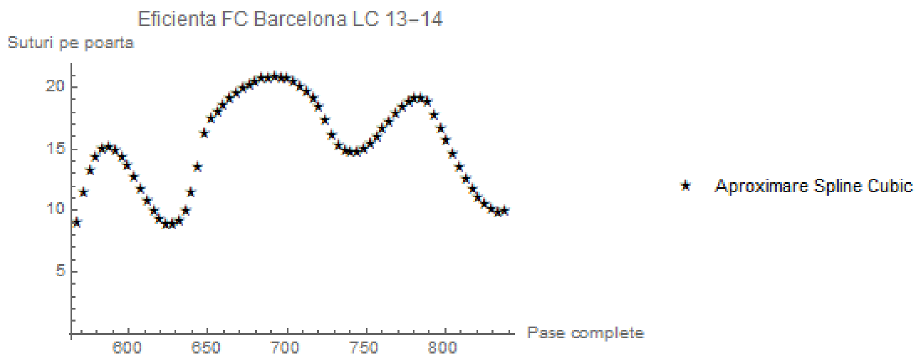


Trecem la crearea punctelor intermediare pentru interpolarea cu functii de imbinare cubice.

```
functieSplineCubicBarcelona = Interpolation[
    suportBarcelona,
    InterpolationOrder->3][x]
]

puncteSplineCubicBarcelona = Table[
    functieSplineCubicBarcelona,
    {x, 568, 838, 4}
]

plotSplineCubicBarcelona = ListPlot[
    puncteSplineCubicBarcelona,
    DataRange -> {568, 838},
    AxesOrigin -> {567, 0},
```

```
    AxesLabel->{"Pase complete", "Suturi pe poarta"},
    PlotLabel->"Eficienta FC Barcelona LC 13-14",
    PlotStyle->Black,
    PlotLegends -> {"Aproximare Spline Cubic"},
    PlotMarkers -> {"\[FivePointedStar]"}
]
```

Graficul interpolarii spline cu functii de imbinare cubice este urmatorul.



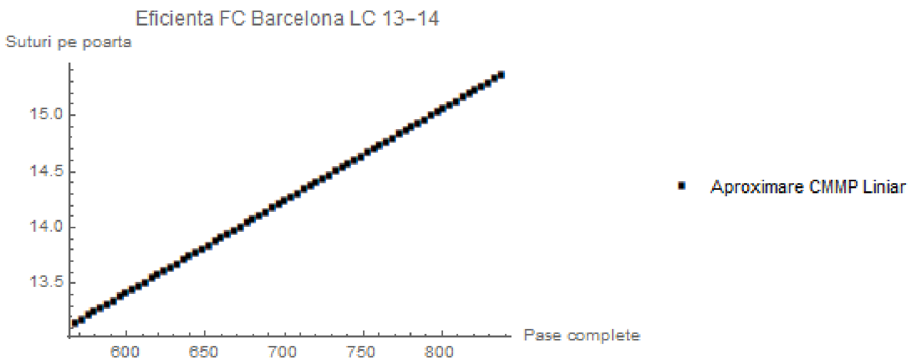
Aproximari in sensul CMMP

In continuare, ne vom ocupa de aproximarea in sensul celor mai mici patrate a suportului interpolarii. Aproximarea liniara se va realiza astfel.

```
functieCMMPLiniarBarcelona = Fit[suportBarcelona, {1, x}, x]

puncteCMMPLiniarBarcelona = Table[
    functieCMMPLiniarBarcelona,
    {x, 568, 838, 4}
]

plotCMMPLiniarBarcelona = ListPlot[
    puncteCMMPLiniarBarcelona,
```

```
    DataRange -> {568, 838},
    AxesOrigin -> {567, 0},
    AxesLabel->{"Pase complete", "Suturi pe poarta"},
    PlotLabel->"Eficienta FC Barcelona LC 13-14",
    PlotStyle->Black,
    PlotLegends -> {"Aproximare CMMP Liniar"},
    PlotMarkers -> {"\[FilledSquare]"}
]
```

Graficul aproximarii liniare in sensul celor mai mici patrate arata in felul urmator. Acesta tinde sa isi creasca numarul de suturi spre poarta daca numarul de pase complete creste, ceea ce pare firesc din punct de vedere fizic.

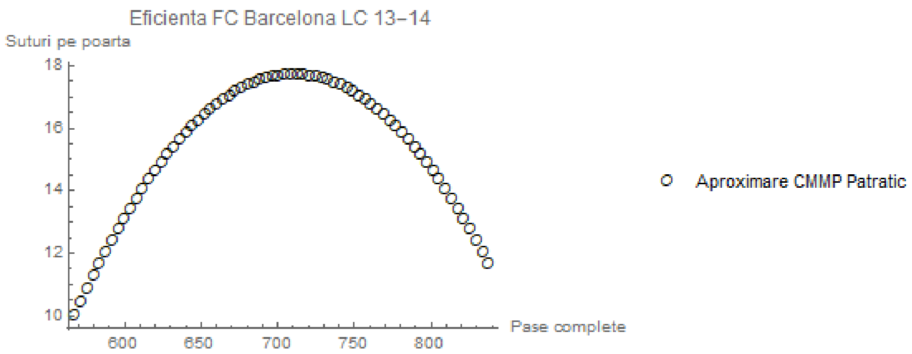


Ne propunem sa gasim si aproximarea patratica in sensul celor mai mici patrate fiindca aceasta poate sa descrie mai bine o eventuala curba (nu o functie liniara) pe care o respecta valorile reale di

```
functieCMMPPatraticBarcelona = Fit[suportBarcelona, {1, x, x^2},
x]
puncteCMMPPatraticBarcelona = Table[
    functieCMMPPatraticBarcelona,
    {x, 568, 838, 4}
]
```

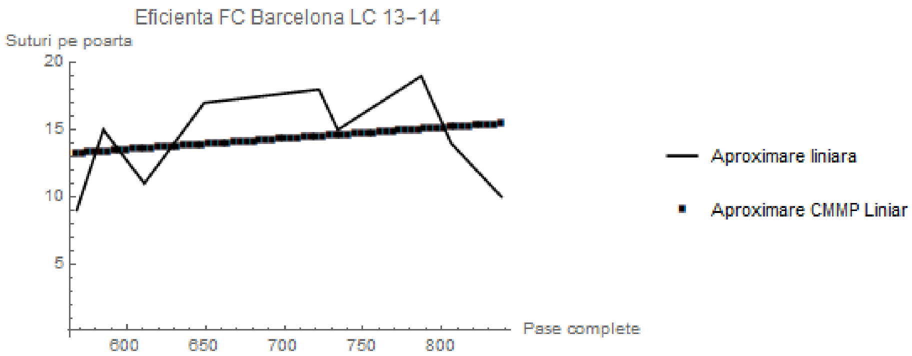
```
plotCMMPPatraticBarcelona = ListPlot[
     puncteCMMPPatraticBarcelona,
     DataRange -> {568, 838},
     AxesOrigin -> {567, 0},
     AxesLabel->{"Pase complete", "Suturi pe poarta"},
     PlotLabel->"Eficienta FC Barcelona LC 13-14",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare CMMP Patratic"},
     PlotMarkers -> {"O"}
]
```

Graficul aproximarii patratice va arata ca cel de mai jos. FC Barcelona, fiind o echipa care paseaza foarte mult, arata slabiciuni in momentul in care paseaza purin sau prea mult negasind cai directe spre poarta adversa, dupa cum se poate observa si in grafic.



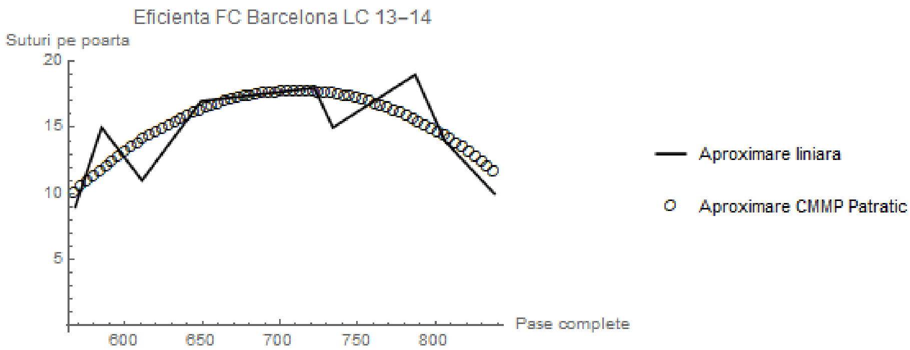
Afisam interpolarea liniara si aproximarea liniara in sensul celor mai mici patrate si observam ca cea de-a doua este aproape orizontala, ceea ce inseamna ca numarul de suturi spre poarta ale echipei FC Barcelona pe parcursul sezonului 2013-2014 a fost in jurul numarului 15, cu mici deviatii.

```
Show[plotLiniarBarcelona, plotCMMPLiniarBarcelona]
```



Continuam cu suprapunerea interpolarii liniare si a aproximarii patratice in sensul celor mai mici patrate.

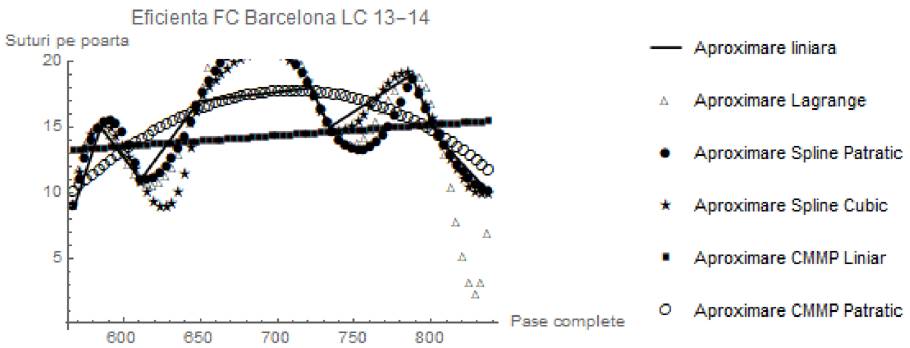
```
Show[plotLiniarBarcelona, plotCMMPLiniarBarcelona]
```



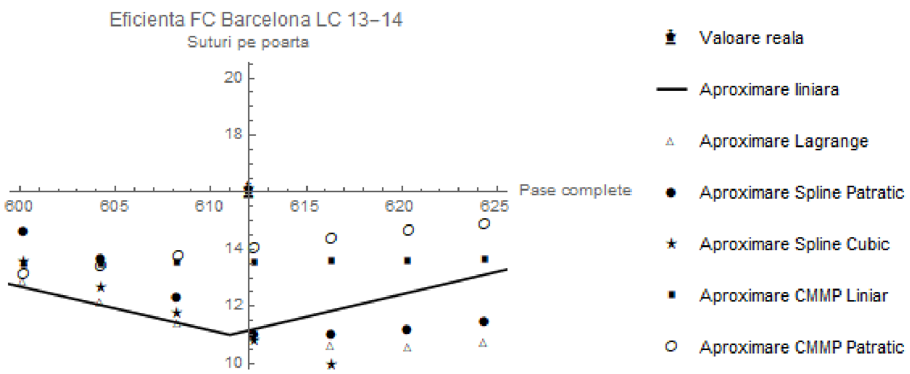
Dorim sa observam toate metodele de interpolare si aproximare folosite, iar apoi, centrand in valoarea reala a meciului ale carui statistici incercam sa le estimam, sa alegem metodele pe care are sens sa le folosim in obtinerea aproximarii.

```
Show[
    plotLiniarBarcelona,
```

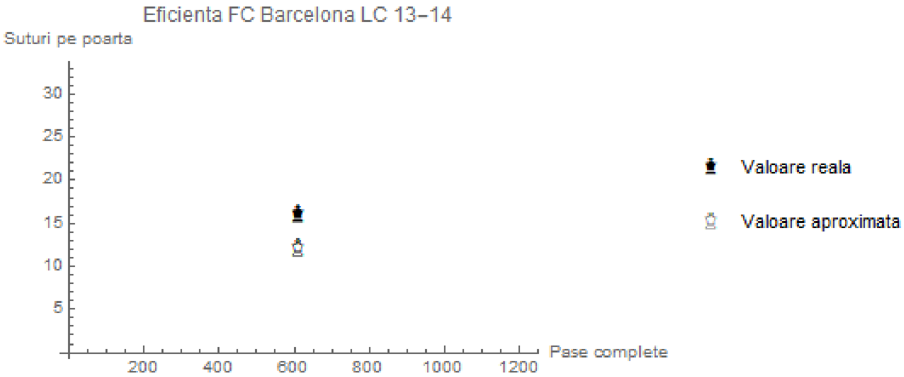
```
    plotLagrBarcelona,
    plotSplinePatraticBarcelona,
    plotSplineCubicBarcelona,
    plotCMMPLiniarBarcelona,
    plotCMMPPatraticBarcelona,
    PlotRange -> {{600, 625}, {10, 20}}
    AxesOrigin -> {612, 16}
]
```



Graficele centrate in punctul {612, 16} arata ca in figura de mai jos.



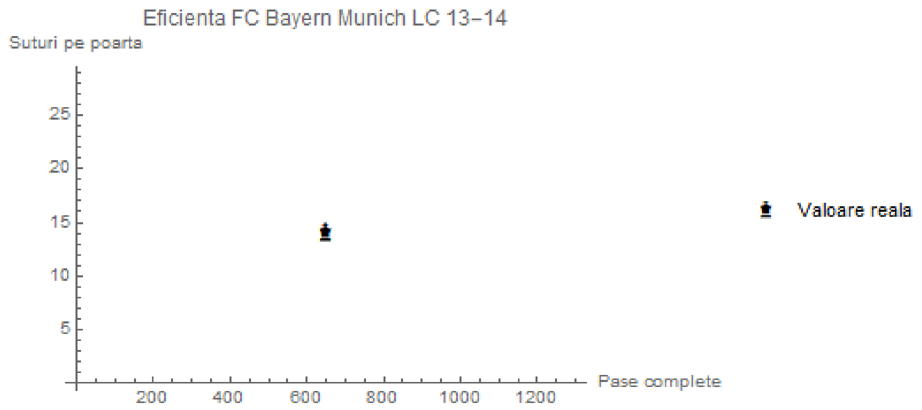
Deoarece toate metodele sunt in vecinatatea valorii reale, le vom folosi pe toate pentru a aproxima valoarea cautata si vor rezulta 12 suturi spre poarta, la diferenta de 4 unitati fata de valoarea reala.



2.2.3 FC Bayern Munich

Acum ne propunem sa aproximam numarul de suturi spre poarta raportate la numarul de pase complete pentru echipa FC Bayern Munich. Aceasta a fost semifinalista, jucand astfel douasprezece meciuri in total in sezonul competitional 2013-2014 al Ligii Campionilor. Vom folosi datele de la toate meciurile in afara in afara de al doilea de dupa faza grupelor, FC Bayern Munich - FC Arsenal 1 - 1, pentru care vom incerca sa aproximam statistica dorita, unde bavarezii au avut 650 de pase complete si 14 suturi spre poarta.

Valoarea reala, ($\{650, 14\}$), arata ca in graficul urmator.



Suportul interpolarii

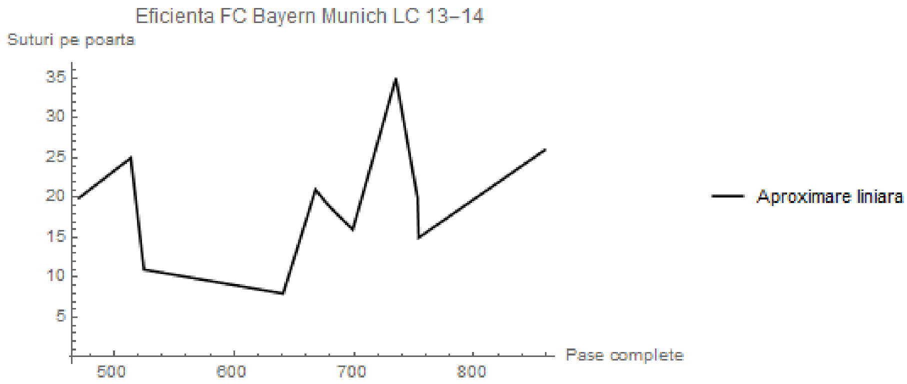
Pentru suportul interpolarii vom folosi statisticile de la toate meciurile jucate in acest sezon al Ligii Campionilor in afara de al doilea meci de dupa faza grupelor, ale carui statistici dorim sa le aproximam.

```
suportBayern = {{471, 20}, {514, 25}, {525, 11}, {641, 8},
{668, 21}, {679, 19}, {699, 16}, {735, 35}, {753, 20}, {754,
15}, {859, 26}}
```

Interpolare liniara

Incepem cu interpolarea liniara pentru a descoperi un eventual model matematic la care tinde graficul.

```
plotLiniarBayern = ListPlot[
    suportBayern,
    AxesOrigin->{470, 0},
    AxesLabel->{"Pase complete", "Suturi pe poarta"},
    PlotLabel->"Eficienta FC Bayern Munich LC 13-14",
    Joined->True,
    PlotStyle->Black,
    PlotLegends->{"Aproximare liniara"}]
```



Interpolare Lagrange

Continuam cu calcularea polinomului de interpolare Lagrange care va avea grad maxim 10 deoarece sunt 11 puncte care il formeaza.

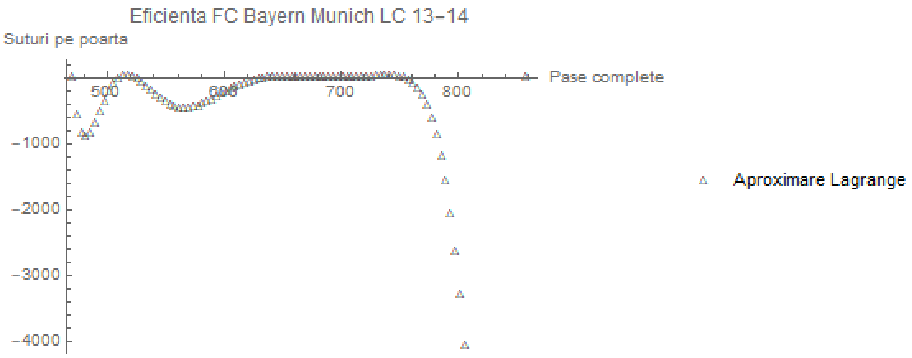
```
functieLagrangeBayern = InterpolatingPolynomial[
     suportBayern,
     x
]
```

Alegem sa cream puncte intermediare intre valorile 471 si 859 adica primul si ultimul punct din suportul interpolarii deoarece nu ne intereseaza ce se afla in afara intervalului.

```
puncteLagrangeBayern = Table[
     functieLagrangeBayern,
     {x, 471, 859, 4}
]

plotLagrangeBayern = ListPlot[
     puncteLagrangeBayern,
     DataRange -> {471, 859},
```

```
    AxesOrigin -> 470, 0,
    AxesLabel->{"Pase complete", "Suturi pe poarta"},
    PlotLabel->"Eficienta FC Bayern Munich LC 13-14",
    PlotStyle->Black,
    PlotLegends -> {"Aproximare Lagrange"},
    PlotMarkers -> {"\[EmptyUpTriangle]"}
]
```

Graficul reprezentativ interpolarii Lagrange este urmatorul.



Observam ca polinomul gasit are foarte multe valori negative care nu pot fi folosite, numarul de suturi spre poarta fiind un numar natural, osciland puternic in capatul superior al suportului de interpolare.

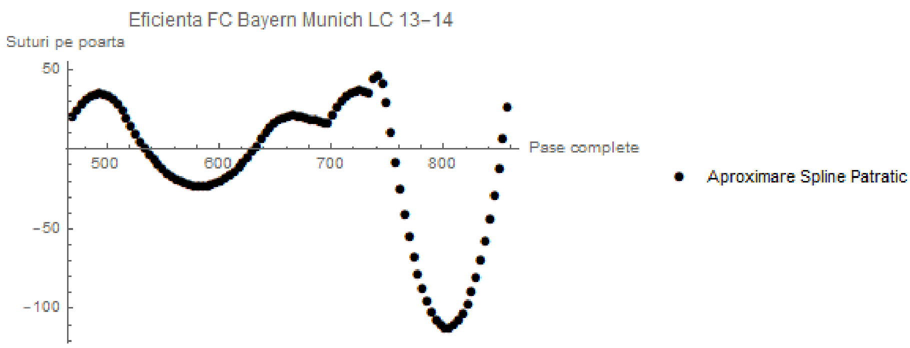
Interpolari de tip spline

Acum vom calcula interpolarile de tip spline. Incepem cu interpolare spline care foloseste functii de imbinare de gradul al doilea. Abordarea in Wolfram Mathematica este urmatoarea.

```
functieSplinePatraticBayern = Interpolation[
    suportBayern,
    InterpolationOrder->2][x]
]
```

```
puncteSplinePatraticBayern = Table[
     functieSplinePatraticBayern,
     {x, 471, 859, 4}
]

plotSplinePatraticBayern = ListPlot[
     puncteSplinePatraticBayern,
     DataRange -> {471, 859},
     AxesOrigin -> {470, 0},
     AxesLabel->{"Pase complete", "Suturi pe poarta"},
     PlotLabel->"Eficienta FC Bayern Munich LC 13-14",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare Spline Patratic"},
     PlotMarkers -> {"\[FilledCircle]"}
]
```

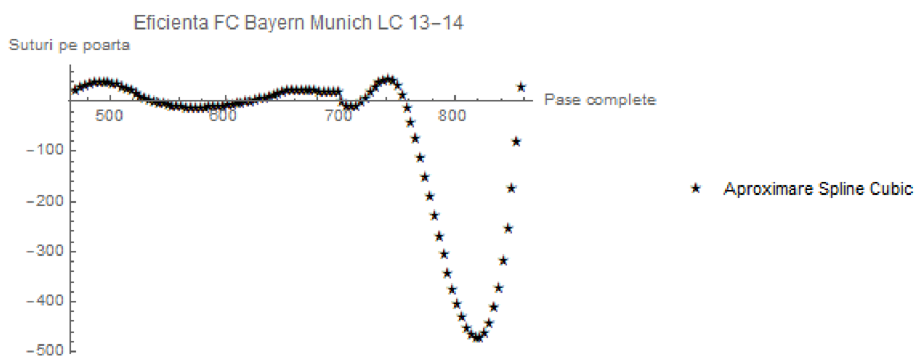
Graficul interpolarii spline cu functii de imbinare patratice arata ca in figura de mai jos.



Urmeaza, in mod analog, calcularea interpolarii spline care foloseste functii de imbinare cubice.

```
functieSplineCubicBayern = Interpolation[
     suportBayern,
```

```
    InterpolationOrder->3][x]
]

puncteSplineCubicBayern = Table[
    functieSplineCubicBayern,
    {x, 471, 859, 4}
]

plotSplineCubicBayern = ListPlot[
    puncteSplineCubicBayern,
    DataRange -> {471, 859},
    AxesOrigin -> {470, 0},
    AxesLabel->{"Pase complete", "Suturi pe poarta"},
    PlotLabel->"Eficienta FC Bayern Munich LC 13-14",
    PlotStyle->Black,
    PlotLegends -> {"Aproximare Spline Cubic"},
    PlotMarkers -> {"\[FivePointedStar]"}
]
```

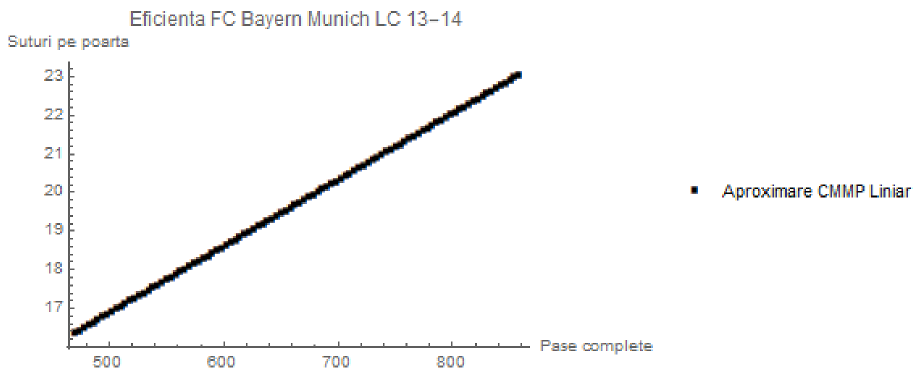


Observam ca desi ambele abordari cu spline-uri au oscilatii, acestea sunt mult mai mici decat in cazul interpolarii Lagrange (-100 pentru functii patratice sau -500 pentru functii cubice in comparatie cu -4000 pentru polinomul Lagrange).

Aproximari in sensul CMMP

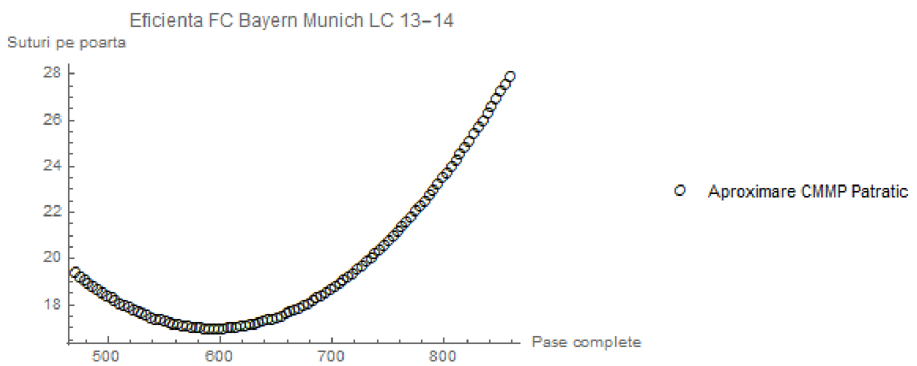
Urmatoarea abordare este aproximarea in sensul celor mai mici pătrate. Incepem cu cea care creeaza un polinom liniar care trece printre toate punctele suportului de interpolare.

```
functieCMMPLiniarBayern = Fit[suportBayern, {1, x}, x]

puncteCMMPLiniarBayern = Table[
     functieCMMPLiniarBayern,
     {x, 471, 859, 4}
]

plotCMMPLiniarBayern = ListPlot[
     puncteCMMPLiniarBayern,
     DataRange -> {471, 859},
     AxesOrigin -> {470, 0},
     AxesLabel->{"Pase complete", "Suturi pe poarta"},
     PlotLabel->"Eficienta FC Bayern Munich LC 13-14",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare CMMP Liniar"},
     PlotMarkers -> {"\[FilledSquare]"}
]
```



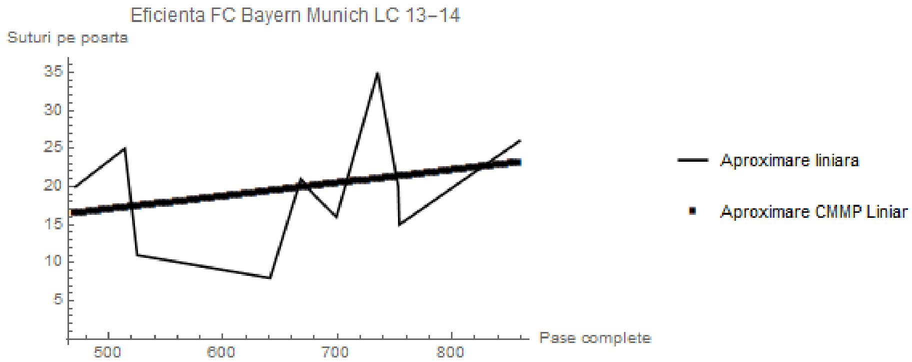
In continuare vom calcula si polinomul patratic care trece printre toate punctele din suportul interpolarii. Acesta se calculeaza in mod analog, folosind aceleasi functii din Wolfram Mathematica.

```
functieCMMPPatraticBayern = Fit[suportBayern, {1, x, x^2}, x]

puncteCMMPPatraticBayern = Table[
     functieCMMPPatraticBayern,
     {x, 471, 859, 4}
]

plotCMMPPatraticBayern = ListPlot[
     puncteCMMPPatraticBayern,
     DataRange -> {471, 859},
     AxesOrigin -> {470, 0},
     AxesLabel->{"Pase complete", "Suturi pe poarta"},
     PlotLabel->"Eficienta FC Bayern Munich LC 13-14",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare CMMP Patratic"},
     PlotMarkers -> {"O"}
]
```

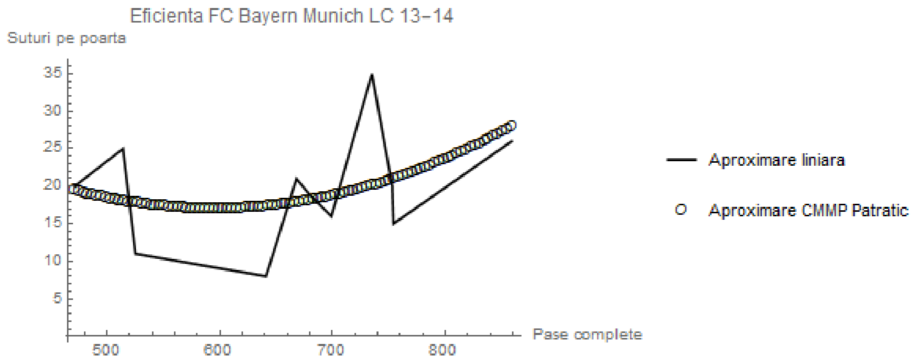


Acum ca am obtinut cele doua grafice, le vom suprapune peste graficul interpolarii liniare pentru a observa cum aproximarile in sensul celor mai mici patrate trec exact printre punctele suport ale interpolarii.

```
Show[plotLiniarBayern, plotCMMPLiniarBayern]
```



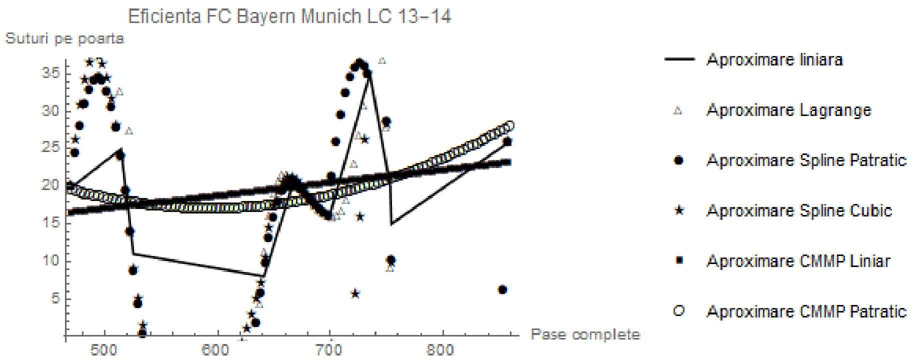

```
Show[plotLiniarBayern, plotCMMPLiniarBayern]
```



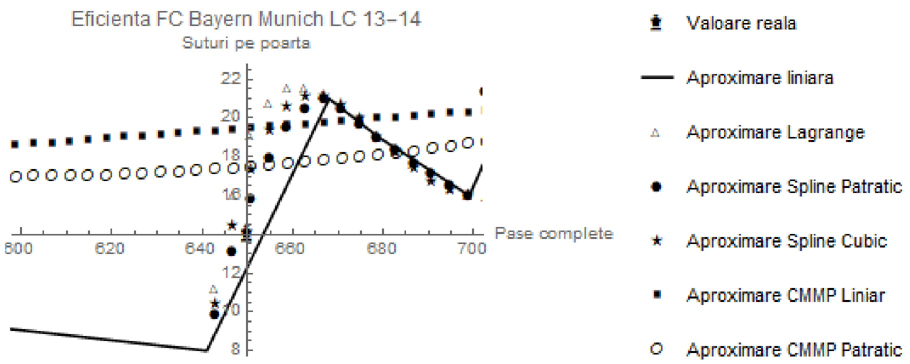

```
Show[
    plotLiniarBayern,
    plotLagrBayern,
    plotSplinePatraticBayern,
    plotSplineCubicBayern,
    plotCMMPLiniarBayern,
    plotCMMPPatraticBayern,
    PlotRange -> {{600, 700}, {8, 22}}
```

```
    AxesOrigin -> {650, 14}
]
```

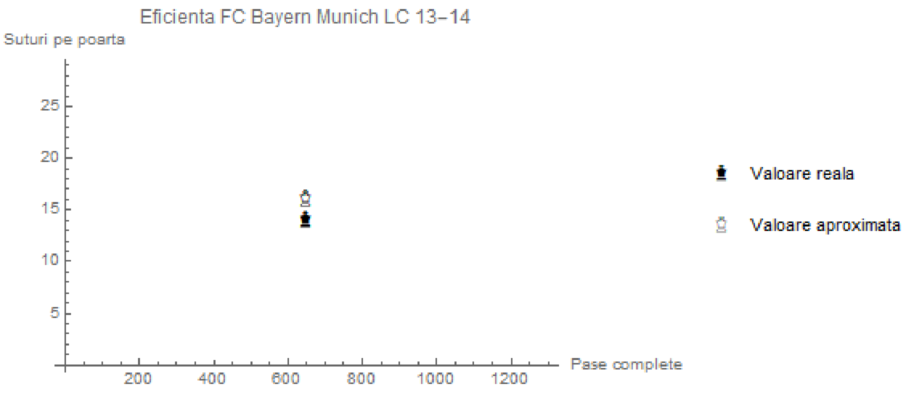
Urmatorul pas este sa afisam toate metodele in acelasi grafic.



Acum, daca centram in punctul pe care dorim sa il aproximam, adica in valoarea reala, $\{650, 14\}$, vom obtine urmatorul grafic.



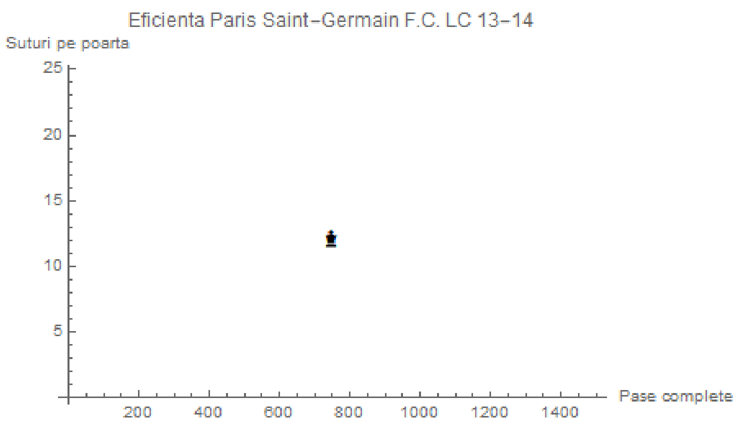
Putem observa ca toate metodele sunt in vecinatatea valorii reale, deci le putem folosi pentru obtinerea estimarii dorite, care va fi $\{650, 16\}$. Diferenta dintre valoarea reala si cea aproximata este de 2, deci am obtinut inca un rezultat foarte bun.



2.2.4 Paris Saint-Germain F.C.

In sezonul 2013-2014 al Ligii Campionilor, Paris Saint-German F.C. a ajuns in sferturile competitiei, jucand astfel zece meciuri in total. Ne intereseaza sa aproximam numarul de suturi spre poarta adversa in primul meci de dupa faza grupelor, Bayer 04 Leverkusen - Paris Saint-Germain F.C. 0 - 4. In acest meci PSG a avut 749 de pase complete si 12 suturi spre poarta.

Acest rezultat se poate vedea in graficul de mai jos.



Suportul interpolarii

In suportul interpolarii vom avea noua puncte disponibile deoarece pe al zecelea, cel din meciul estimat, nu il putem folosi.

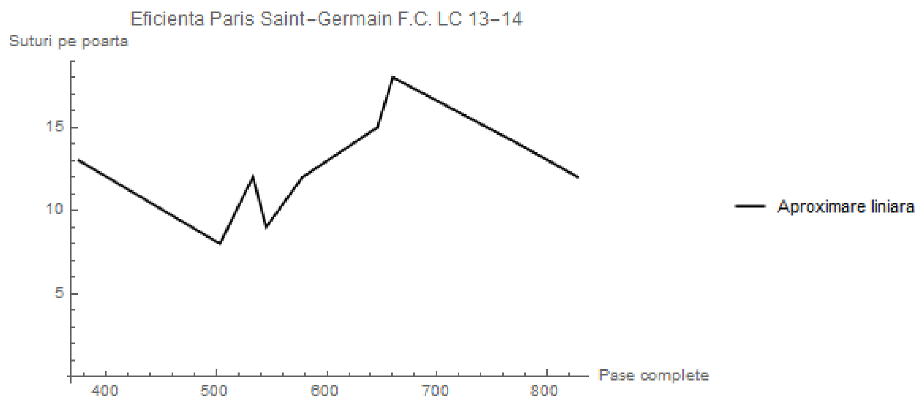
```
suportPSG = {{375, 13}, {503, 8}, {533, 12}, {545, 9}, {578,
12}, {646, 15 {660, 18}, {774, 14}, {828, 12}}
```

Interpolare liniara

Prima abordare este cea in care construim interpolarea liniara.

```
plotLiniarPSG = ListPlot[
     suportPSG,
     AxesOrigin->{374, 0},
     AxesLabel->{"Pase complete", "Suturi pe poarta"},
     PlotLabel->"Eficienta Paris Saint-Germain F.C. LC 13-14",
     Joined->True,
     PlotStyle->Black,
     PlotLegends->{"Aproximare liniara"}
]
```

Graficul interpolarii liniare arata ca in figura de mai jos.



Interpolare Lagrange

In continuare, construim polinomul de interpolare Lagrange care va avea grad maxim 8 deoarece sunt 9 puncte care alcatuiesc suportul de interpolare.

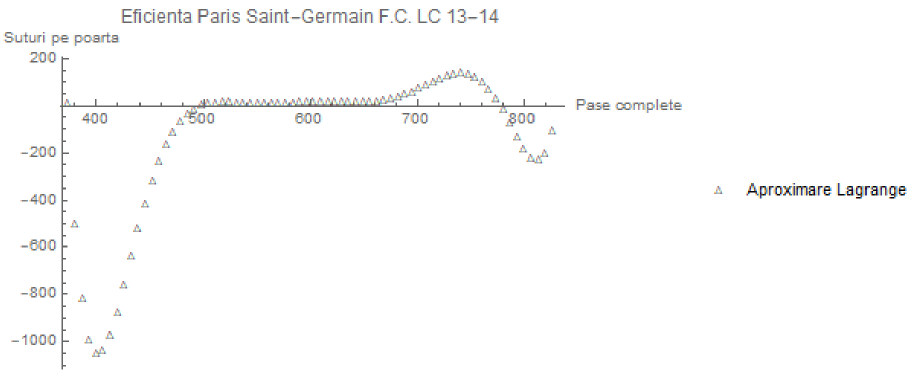
Codul Wolfram Mathematica folosit pentru generarea polinomului, a punctelor intermediare care vor fi plotate si a graficului final este urmatorul.

```
functieLagrangePSG = InterpolatingPolynomial[
     suportPSG,
     x
]
```

Deoarece distanta dintre doua puncte consecutive din suportul interpolarii pe axa paselor complete este mare, vom folosi un pas de 6.5 pentru generarea punctelor intermediare.

```
puncteLagrangePSG = Table[
     functieLagrangePSG,
     {x, 375, 828, 6.5}
]

plotLagrangePSG = ListPlot[
     puncteLagrangePSG,
     DataRange -> {375, 828},
     AxesOrigin -> 374, 0,
     AxesLabel->{"Pase complete", "Suturi pe poarta"},
     PlotLabel->"Eficienta Paris Saint-Germain F.C. LC 13-14",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare Lagrange"},
     PlotMarkers -> {"\[EmptyUpTriangle]"}
]
```

Graficul interpolarii Lagrange este prezentat in figura de mai jos. Se observa, ca si in alte cazuri anterioare, oscilatii mari la capetele suportului interpolarii.



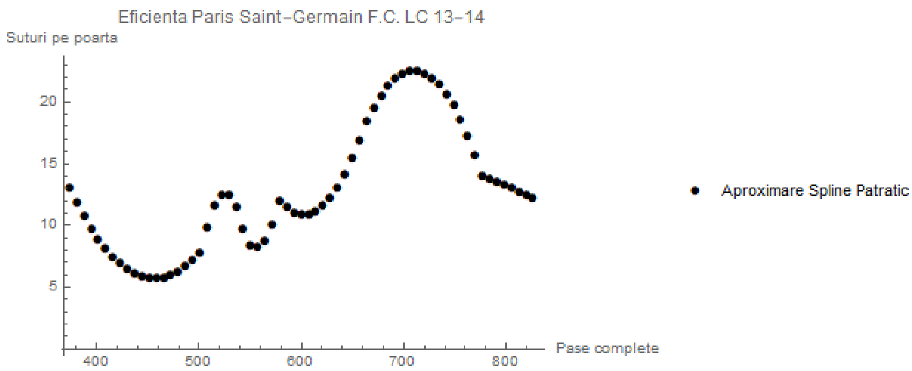
Interpolari de tip spline

Pentru ca gradul polinomului Lagrange este mare, acesta produce si o eroare mare. Pentru a incerca sa micsoram aceasta eroare, continuam cu interpolari de tip spline care isi propun sa nu se foloseasca de la fel de multe date ca polinomul Lagrange, insa intre fiecare doua puncte consecutive sa creeze un polinom de grad 2 sau 3 care sa micsoreze erorile.

```
functieSplinePatraticPSG = Interpolation[
     suportPSG,
     InterpolationOrder->2][x]
]

puncteSplinePatraticPSG = Table[
     functieSplinePatraticPSG,
     {x, 375, 828, 7}
]

plotSplinePatraticPSG = ListPlot[
     puncteSplinePatraticPSG,
     DataRange -> {375, 828},
```

```
    AxesOrigin -> {374, 0},
    AxesLabel->{"Pase complete", "Suturi pe poarta"},
    PlotLabel->"Eficienta Paris Saint-Germain F.C. LC 13-14",
    PlotStyle->Black,
    PlotLegends -> {"Aproximare Spline Patratic"},
    PlotMarkers -> {"\[FilledCircle]"}
]
```

Puteti vedea mai jos graficul interpolarii spline care foloseste functii de imbinare patratice.

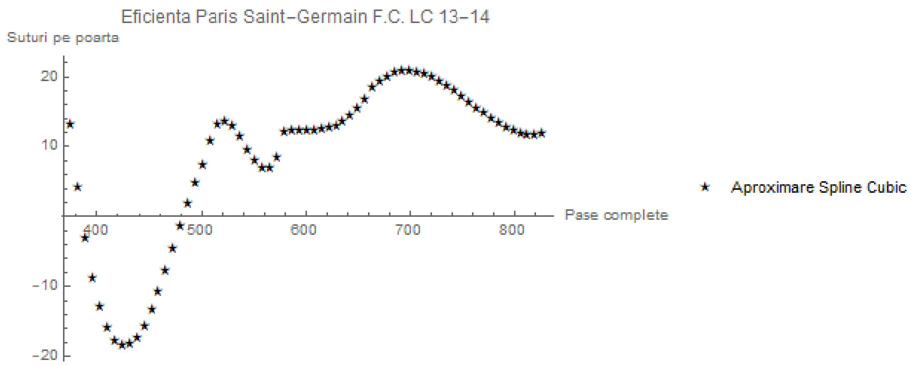


Acum dorim sa calculam functiile de imbinare cubice pentru interpolarea spline.

```
functieSplineCubicPSG = Interpolation[
    suportPSG,
    InterpolationOrder->3][x]

puncteSplineCubicPSG = Table[
    functieSplineCubicPSG,
    {x, 375, 828, 7}
]

plotSplineCubicPSG = ListPlot[
```

```
    puncteSplineCubicPSG,
    DataRange -> {375, 828},
    AxesOrigin -> {374, 0},
    AxesLabel->{"Pase complete", "Suturi pe poarta"},
    PlotLabel->"Eficienta Paris Saint-Germain F.C. LC 13-14",
    PlotStyle->Black,
    PlotLegends -> {"Aproximare Spline Cubic"},
    PlotMarkers -> {"\[FivePointedStar]"}
]
```

Graficul interpolarii spline care foloseste functii de imbinare de grad trei este urmatorul.



Observam ca, spre deosebire de cea care foloseste functii de imbinare de grad doi, aceasta oscileaza in capatul inferior al suportului interpolarii si aproximeaza cateva valori negative care ar trebui eliminate in caz ca valoarea reala s-ar afla in acea vecinatate.

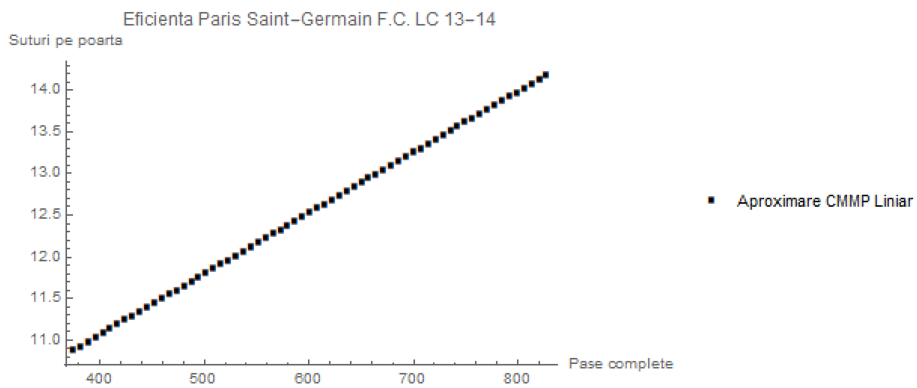
Aproximari in sensul CMMP

In continuare, ne ocupam de aproximarile liniara si patratica in sensul celor mai mici patrate. Incepem cu aproximarea printr-un polinom liniar.

```
functieCMMPLiniarPSG = Fit[suportPSG, {1, x}, x]
```

```
puncteCMMPLiniarPSG = Table[
     functieCMMPLiniarPSG,
     {x, 375, 828, 7}
]

plotCMMPLiniarPSG = ListPlot[
     puncteCMMPLiniarPSG,
     DataRange -> {375, 828},
     AxesOrigin -> {374, 0},
     AxesLabel->{"Pase complete", "Suturi pe poarta"},
     PlotLabel->"Eficienta Paris Saint-Germain F.C. LC 13-14",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare CMMP Liniar"},
     PlotMarkers -> {"\[FilledSquare]"}
]
```

Graficul aproximarii liniare in sensul celor mai mici patrate a punctelor care apartin suportului interpolarii se poate observa mai jos.

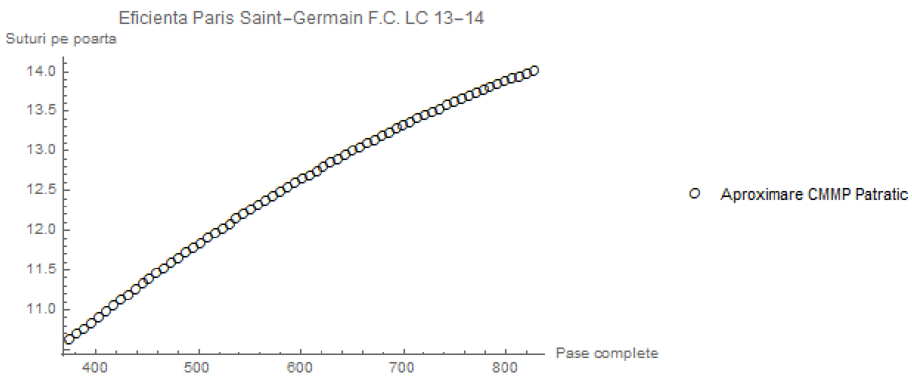


Continuam cu gasirea unui polinom patratic care sa minimizeze suma distantelor fata de punctele din suport.

```
functieCMMPPatraticPSG = Fit[suportPSG, {1, x, x^2}, x]
```

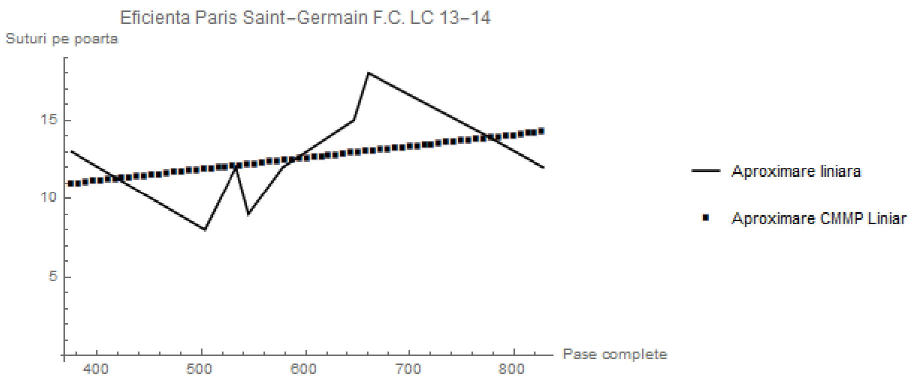
```
puncteCMMPPatraticPSG = Table[
     functieCMMPPatraticPSG,
     {x, 375, 828, 7}
]

plotCMMPPatraticPSG = ListPlot[
     puncteCMMPPatraticPSG,
     DataRange -> {375, 828},
     AxesOrigin -> {374, 0},
     AxesLabel->{"Pase complete", "Suturi pe poarta"},
     PlotLabel->"Eficienta Paris Saint-Germain F.C. LC 13-14",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare CMMP Patratic"},
     PlotMarkers -> {"O"}
]
```

Aproximarea patratica in sensul celor mai mici patrate arata astfel.

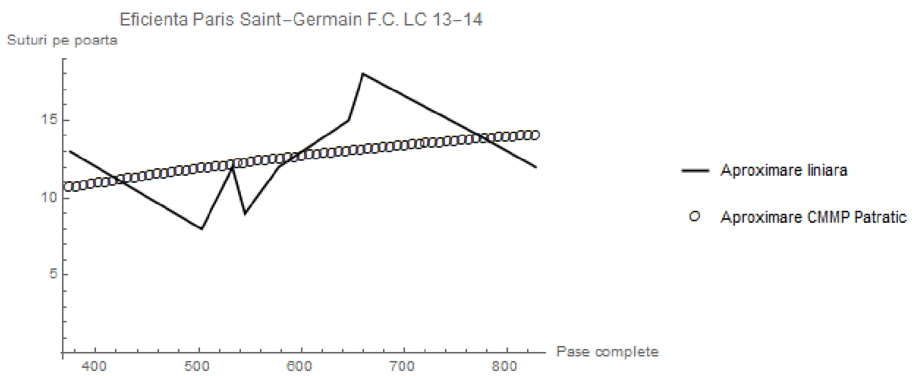


Urmeaza sa afisam suprapunerea dintre interpolarea liniara si cele doua aproximari in sensul celor mai mici patrate.

```
Show[plotLiniarPSG, plotCMMPLiniarPSG]
```



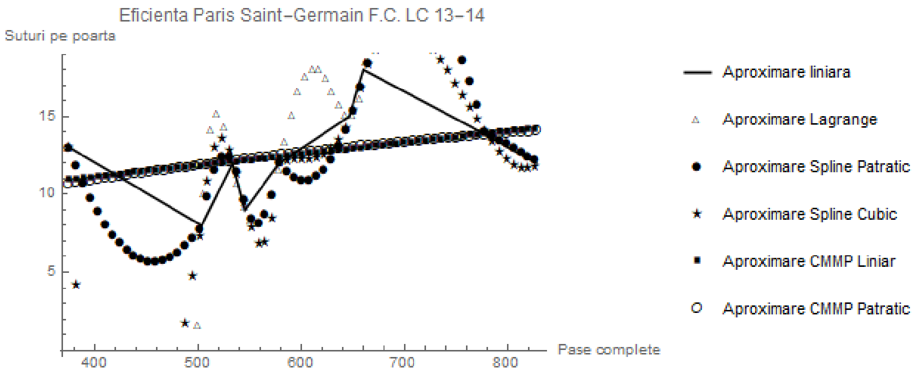

```
Show[plotLiniarPSG, plotCMMPLiniarPSG]
```



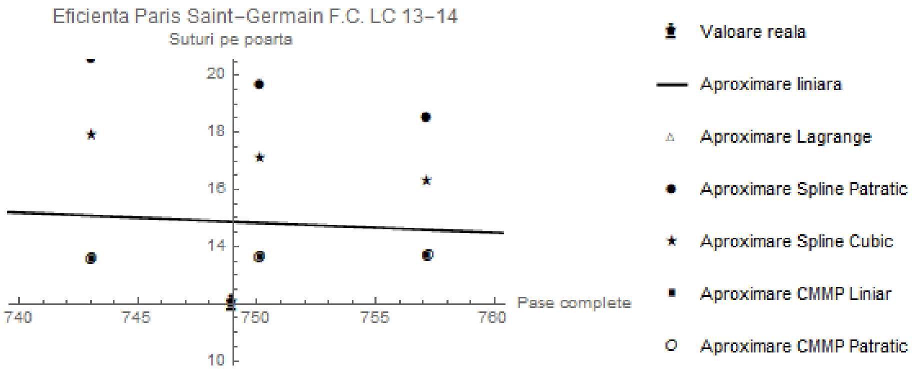
Inainte sa gasim valoarea pe care dorim sa o estimam, afisam rezultatele tuturor metodelor aplicate in acelasi grafic.

```
Show[
    plotLiniarPSG,
    plotLagrPSG,
    plotSplinePatraticPSG,
    plotSplineCubicPSG,
    plotCMMPLiniarPSG,
    plotCMMPPatraticPSG,
    PlotRange -> {{740, 760}, {10, 20}}
```

```
    AxesOrigin -> {749, 12}
]
```

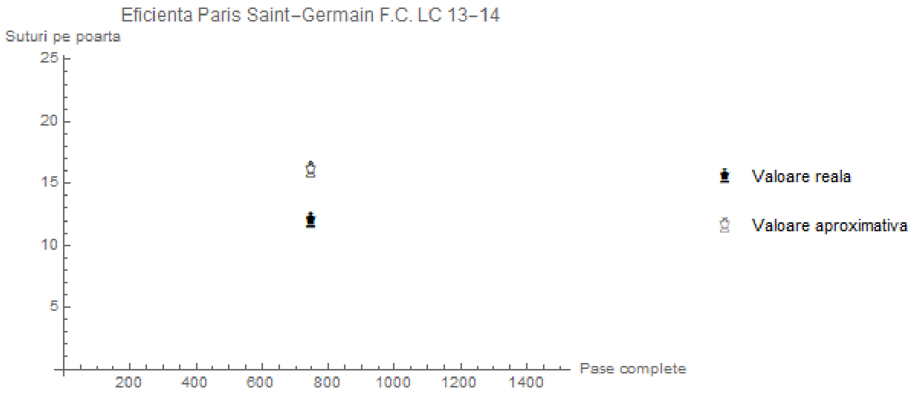


Centram graficul in valoarea reala pentru a vedea cate dintre metodele calculate pot fi folosite cu succes in a calcula valoarea aproximata.



Observam ca interpolarea Lagrange nu se afla in vecinatatea valorii reale, deci nu are sens sa o consideram pentru calcularea valorii aproximate, care va fi evaluata la $\{749, 16\}$.

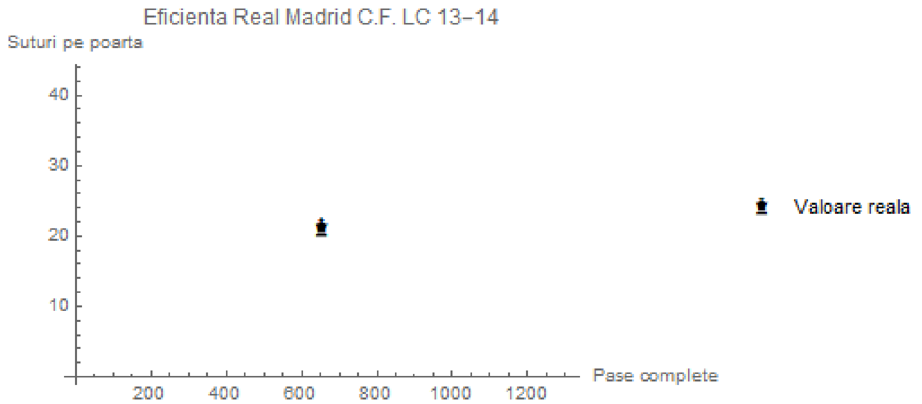
Diferenta dintre valoarea reala si cea aproximata arata ca in figura de mai jos. Aceasta este de 4 unitati, reprezentand un rezultat nu exceptional, dar bun.



2.2.5 Real Madrid C.F.

Ne ocupam de data aceasta de aproximarea statisticilor echipei castigatoare din sezonul competitional 2013-2014, Real Madrid C.F. Aceasta a jucat in total treisprezece meciuri, iar pe noi ne intereseaza sa estimam numarul de pase complete si numarul de suturi spre poarta din finala, Real Madrid C.F. - Atletico Madrid 4 - 1.

Valoarea reala a acestei estimari este reprezentata prin punctul {655, 21} in graficul de mai jos.



Suportul interpolarii

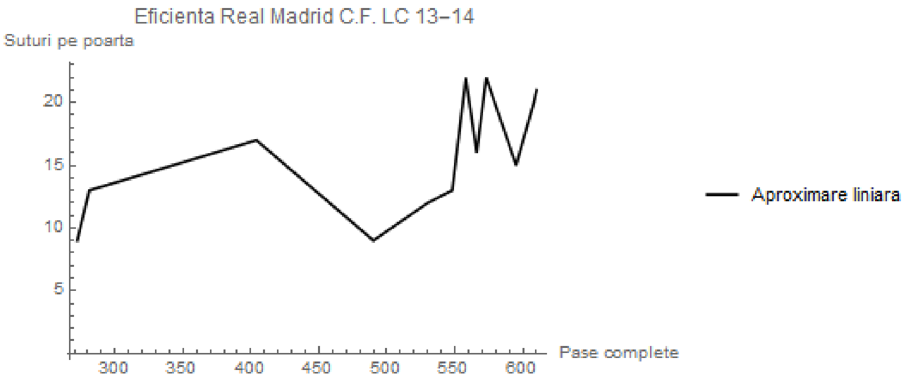
Suportul interpolarii va fi format din statisticile tuturor meciurilor in afara de meciul final.

```
suportReal = {{272, 9}, {281, 13}, {404, 17}, {490, 9}, {530,
12}, {548, 13}, {558, 22}, {566, 16}, {573, 22}, {595, 15},
{608, 20}, {610, 21}}
```

Interpolare liniara

Primul tip de interpolare pe care o aplicam asupra suportului este interpolarea liniara. Aceasta arata ca in imaginea urmatoare.

```
plotLiniarReal = ListPlot[
     suportReal,
     AxesOrigin->{271, 0},
     AxesLabel->{"Pase complete", "Suturi pe poarta"},
     PlotLabel->"Eficienta Real Madrid C.F. LC 13-14",
     Joined->True,
     PlotStyle->Black,
     PlotLegends->{"Aproximare liniara"}
]
```

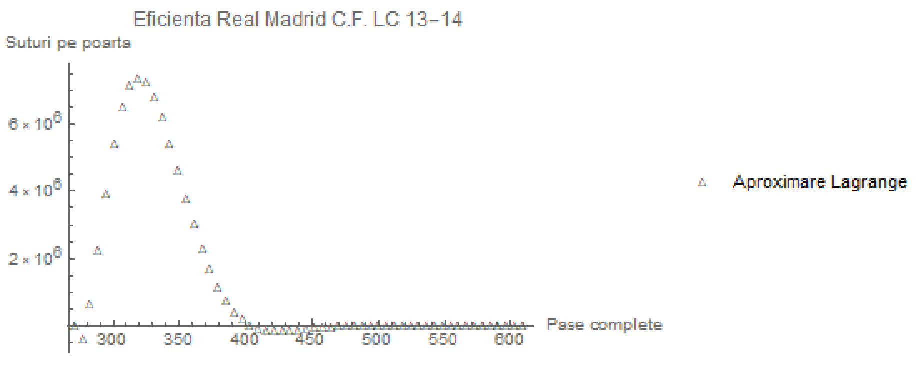


Interpolare Lagrange

Interpolarea Lagrange va fi realizata prin codul Wolfram Mathematica urmator.

```
functieLagrangeReal = InterpolatingPolynomial[
     suportReal,
     x
]
puncteLagrangeReal = Table[
     functieLagrangeReal,
     {x, 272, 610, 6}
]
plotLagrangeReal = ListPlot[
     puncteLagrangeReal,
     DataRange -> {272, 610},
     AxesOrigin -> 271, 0,
     AxesLabel->{"Pase complete", "Suturi pe poarta"},
     PlotLabel->"Eficienta Real Madrid C.F. LC 13-14",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare Lagrange"},
     PlotMarkers -> {"\[EmptyUpTriangle]"}
]
```

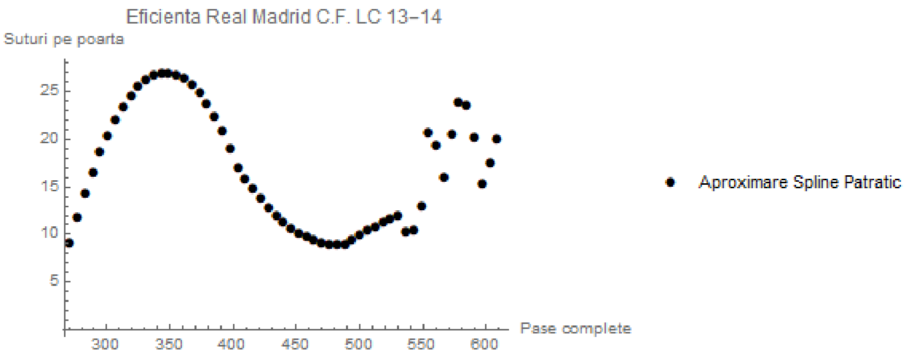
Observam oscilatii enorme in capatul inferior al suportului interpolarii.



Interpolari de tip spline

Urmeaza interpolari de tip spline cu functii de imbinare de gradul doi.

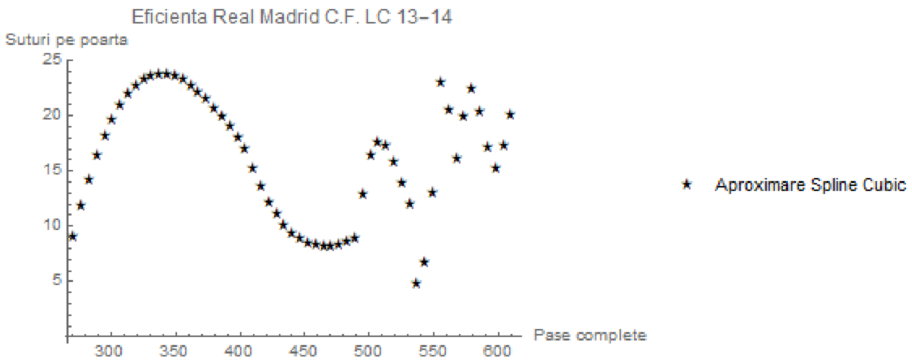
```
functieSplinePatraticReal = Interpolation[
     suportReal,
     InterpolationOrder->2][x]
]
puncteSplinePatraticReal = Table[
     functieSplinePatraticReal,
     {x, 272, 610, 6}
]
plotSplinePatraticReal = ListPlot[
     puncteSplinePatraticReal,
     DataRange -> {272, 610},
     AxesOrigin -> {271, 0},
     AxesLabel->{"Pase complete", "Suturi pe poarta"},
     PlotLabel->"Eficienta Real Madrid C.F. LC 13-14",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare Spline Patratic"},
     PlotMarkers -> {"\[FilledCircle]"}
]
```



Observam ca spre marginea superioara a graficului, unde este o aglomerare de puncte cunoscute, avand mai multe functii de tip spline, graficul va fi mai neregulat.

In continuare, ne ocupam de interpolarea de tip spline care foloseste functii de imbinare cubice.

```
functieSplineCubicReal = Interpolation[
     suportReal,
     InterpolationOrder->3][x]

puncteSplineCubicReal = Table[
     functieSplineCubicReal,
     {x, 272, 610, 6}
]

plotSplineCubicReal = ListPlot[
     puncteSplineCubicReal,
     DataRange -> {272, 610},
     AxesOrigin -> {271, 0},
     AxesLabel->{"Pase complete", "Suturi pe poarta"},
     PlotLabel->"Eficienta Real Madrid C.F. LC 13-14",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare Spline Cubic"},
     PlotMarkers -> {"\[FivePointedStar]"}
]
```



Se poate observa acelasi comportament ca si la functiile spline patratice generate anterior.

Aproximari in sensul CMMP

Urmeaza aproximarea liniara in sensul celor mai mici patrate.

Suportul de interpolare contine puncte care se afla intr-un interval destul de larg pe axa suturilor spre poarta, deci ne putem astepta ca aproximarea liniara in sensul celor mai mici patrate sa fie reprezentata printr-o dreapta mai inclinata decat in exemplele anterioare.

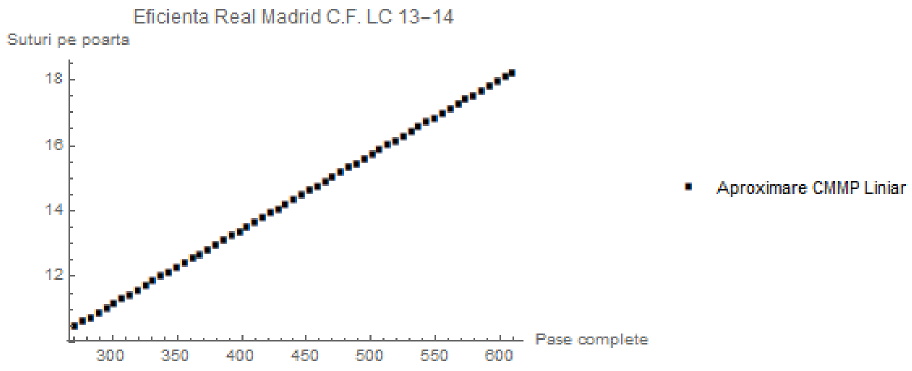
```
functieCMMPLiniarReal = Fit[suportReal, {1, x}, x]

puncteCMMPLiniarReal = Table[
     functieCMMPLiniarReal,
     {x, 272, 610, 6}
]
```

Pentru a putea reprezenta mai bine punctele in Wolfram Mathematica s-a ales un pas egal cu 6 pentru a avea intre 50 si 60 de puncte intre 272 si 610, care reprezinta cel mai mic si cel mai mare numar de pase complete din suportul interpolarii.

Un grafic cu prea multe puncte nu mai poate fi suprapus cu alte grafice, iar unul cu prea putine nu detine destula informatie cat sa ajute la compararea a doua metode de aproximare.

```
plotCMMPLiniarReal = ListPlot[
     puncteCMMPLiniarReal,
     DataRange -> {272, 610},
     AxesOrigin -> {271, 0},
     AxesLabel->{"Pase complete", "Suturi pe poarta"},
     PlotLabel->"Eficienta Real Madrid C.F. LC 13-14",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare CMMP Liniar"},
     PlotMarkers -> {"\[FilledSquare]"}
]
```

Acesta este graficul aproximarii liniare in sensul celor mai mici patrate.

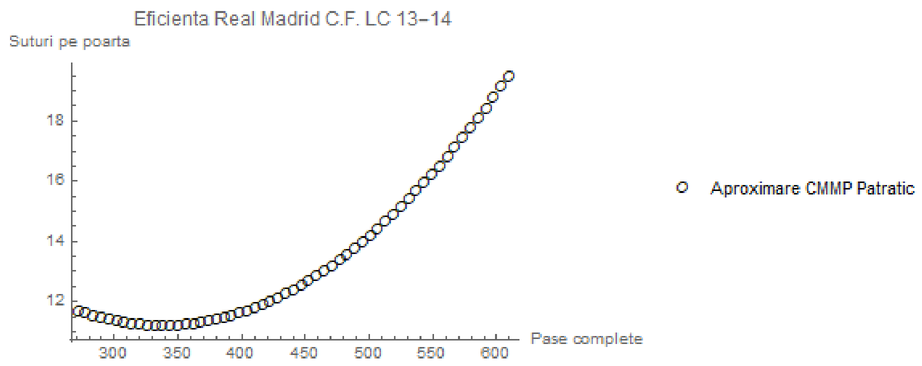


Pentru aproximarea patratica in sensul celor mai mici patrate, rationamentul este analog cu cel de mai sus.

Wolfram Mathematica ne ajuta sa obtinem acest polinom astfel.

```
functieCMMPPatraticReal = Fit[suportReal, {1, x, x^2}, x]

puncteCMMPPatraticReal = Table[
     functieCMMPPatraticReal,
     {x, 272, 610, 6}
]

plotCMMPPatraticReal = ListPlot[
     puncteCMMPPatraticReal,
     DataRange -> {272, 610},
     AxesOrigin -> {271, 0},
     AxesLabel->{"Pase complete", "Suturi pe poarta"},
     PlotLabel->"Eficienta Real Madrid C.F. LC 13-14",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare CMMP Patratic"},
     PlotMarkers -> {"O"}
]
```

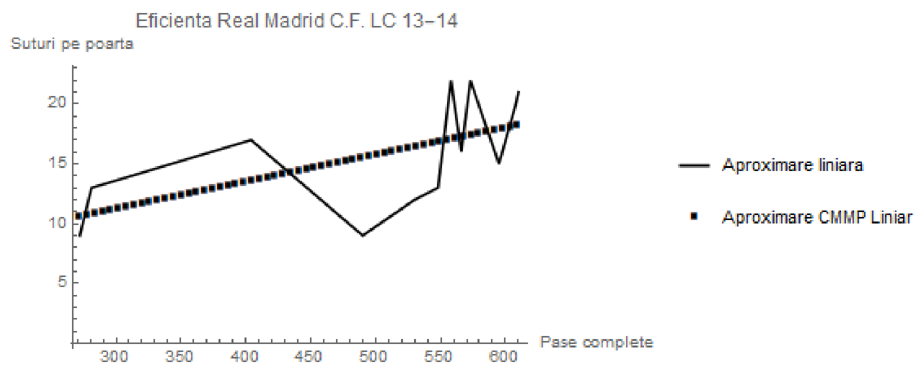
Graficul aproximarii patratice in sensul celor mai mici patrate arata ca mai jos.



Suprapunem interpolarea liniara cu cele doua aproximari in sensul celor mai mici patrate pentru a observa daca acestea trec printre punctele care alcatuiesc suportul interpolarii.

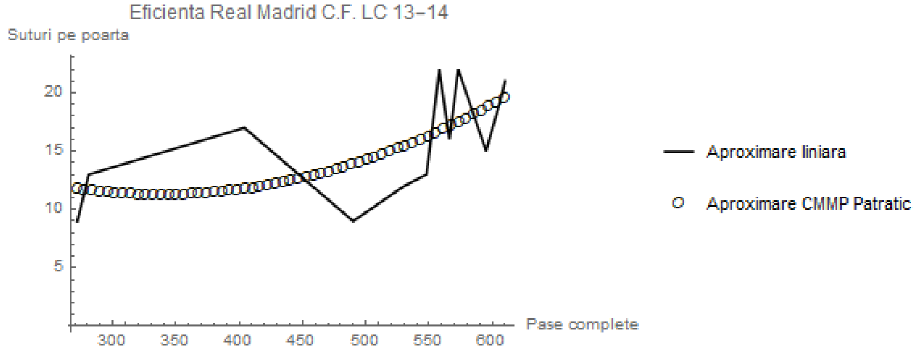
Prima data folosim aproximarea liniara in sensul celor mai mici patrate.

```
Show[plotLiniarReal, plotCMMPLiniarReal]
```



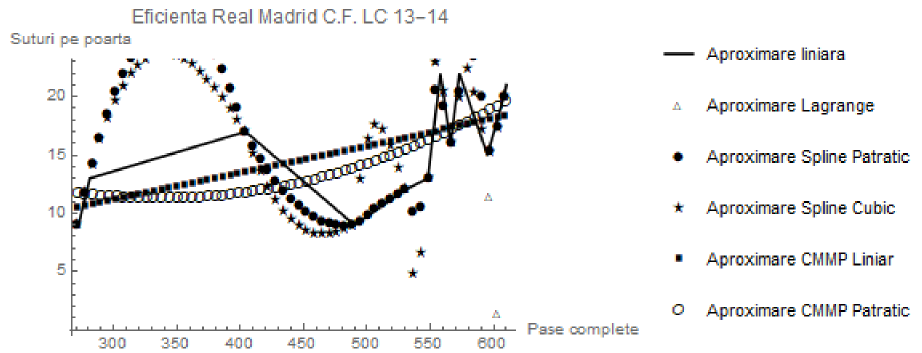
Continuam cu aproximarea patratica in sensul celor mai mici patrate.

```
Show[plotLiniarReal, plotCMMPLiniarReal]
```

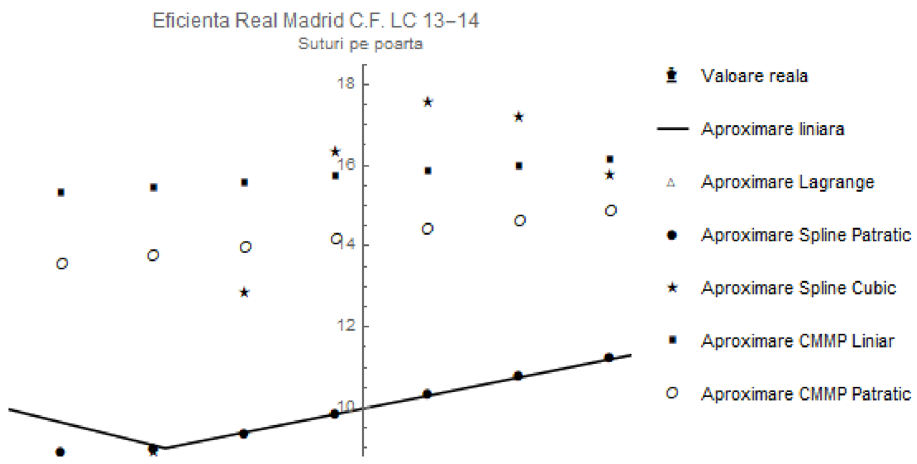


Urmatorul pas este sa suprapunem toate aproximarile gasite si sa le centram in punctul pe care il cautam pentru a elimina eventuale metode care au erori prea mari.

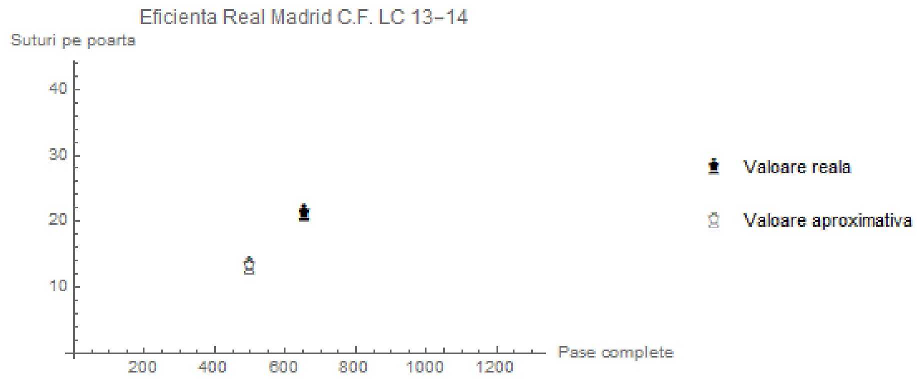
```
Show[
    plotLiniarReal,
    plotLagrReal,
    plotSplinePatraticReal,
    plotSplineCubicReal,
    plotCMMPLiniarReal,
    plotCMMPPatraticReal,
    PlotRange -> {{480, 520}, {9, 18}}
    AxesOrigin -> {503, 8}
]
```



Observam ca interpolarea Lagrange nu este in vecinatatea valorii reale cautate de noi, asa ca nu o vom considera valida, astfel vom face media valorilor celorlalte metode in punctul 655.



Vom aproxima 13 suturi spre poarta pentru 503 pase complete. Diferenta fata de 21, valoarea reala pe care ne doream sa o estimam, se datoreaza si faptului ca in meciul din finala au existat prelungiri, in care Real Madrid C.F. a avut foarte multe ocazii.



2.3 Liga Campionilor 2014-2015

In acest an ne-am propus sa urmarim evolutia suturilor la poarta fata de numarul de faulturi efectuate.

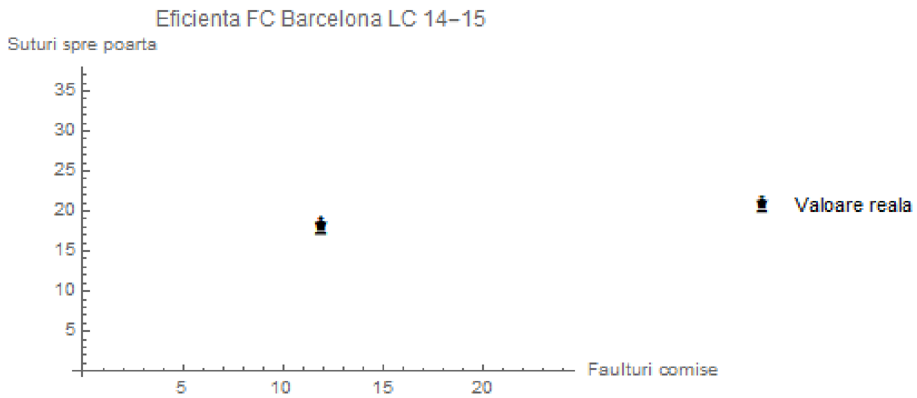
Am selectat urmatoarele echipe:

- FC Barcelona
- FC Bayern Munich
- Chelsea F.C.
- Paris Saint-Germain F.C.
- Real Madrid C.F.

2.3.1 FC Barcelona

Dorim in continuare sa aproximam numarul de suturi spre poarta ale echipei FC Barcelona din finala sezonului competitional 2014-2015 al Ligii Campionilor, respectiv Juventus F.C. - FC Barcelona 1 - 3. In acest meci, Barcelona a avut un numar de 12 faulturi comise si 18 suturi spre poarta.

Ne vom raporta la treisprezece meciuri si valoarea reala pe care dorim sa o aproximam (12, 18) este reprezentata grafic mai jos.



Suportul interpolarii

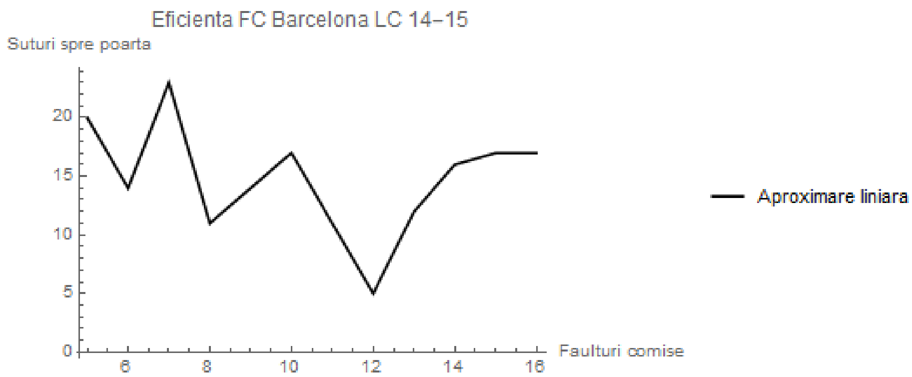
Suportul interpolarii va fi format din datele adunate de la douasprezece meciuri, adica douasprezece perechi de tipul (faulturi comise, suturi spre poarta) sortate dupa numaru de faulturi comise.

```
suportBarcelona = {{5, 20}, {6, 14}, {7, 23}, {8, 11}, {9, 14},
{10, 17}, {11, 11}, {12, 5}, {13, 12}, {14, 16}, {15, 17}, {16,
17}}
```

Interpolare liniara

In continuare dorim sa afisam un grafic reprezentand interpolarea liniara a acestor puncte, aceasta fiind prima modalitate care ne va ajuta sa aproximam numarul de suturi spre poarta din meciul selectat mai sus.

```
plotLiniarBarcelona = ListPlot[
     suportBarcelona,
     AxesOrigin->{4, 0},
     AxesLabel->{"Faulturi comise", "Suturi spre poarta"},
     PlotLabel->"Eficienta FC Barcelona LC 14-15",
     Joined->True,
     PlotStyle->Black,
     PlotLegends->{"Aproximare liniara"}
]
```



Graficul interpolarii liniare este cel de mai sus.

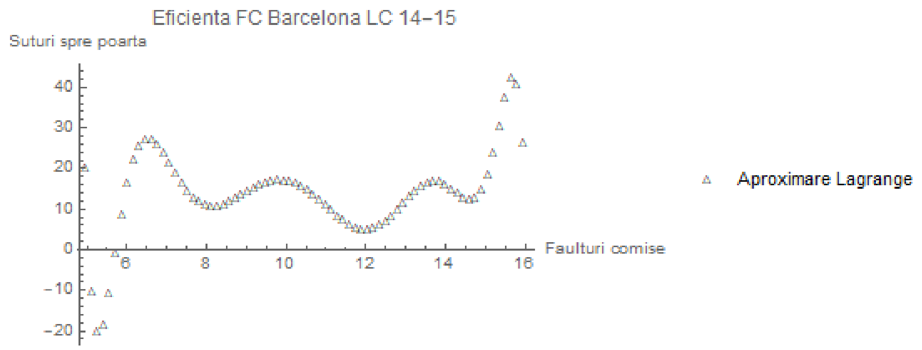
Interpolare Lagrange

Al doilea tip de interpolare pe care o vom folosi se realizeaza cu ajutorul polinomului Lagrange care va fi de grad maxim 11 deoarece folosim 12 puncte in suportul interpolarii.

Modul in care acesta este construit in Wolfram Mathematica este urmatorul.

```
functieLagrangeBarcelona = InterpolatingPolynomial[
     suportBarcelona,
     x
]

puncteLagrangeBarcelona = Table[
     functieLagrangeBarcelona,
     {x, 5, 16, 0.15}
]

plotLagrangeBarcelona = ListPlot[
     puncteLagrangeBarcelona,
     DataRange -> {5, 16},
     AxesOrigin -> 4, 0,
     AxesLabel->{"Faulturi comise", "Suturi spre poarta"},
     PlotLabel->"Eficienta FC Barcelona LC 14-15",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare Lagrange"},
     PlotMarkers -> {"\[EmptyUpTriangle]"}
]
```

Graficul interpolarii Lagrange este cel de mai jos.



Observam ca acesta poate avea si valori negative, lucru care nu poate fi valabil din punct de vedere fizic gandindu-ne la ce dorim noi sa aproximam, adica suturile spre poarta. Acest fapt ne arata cat de sensibil este acest polinom la oscilatii.

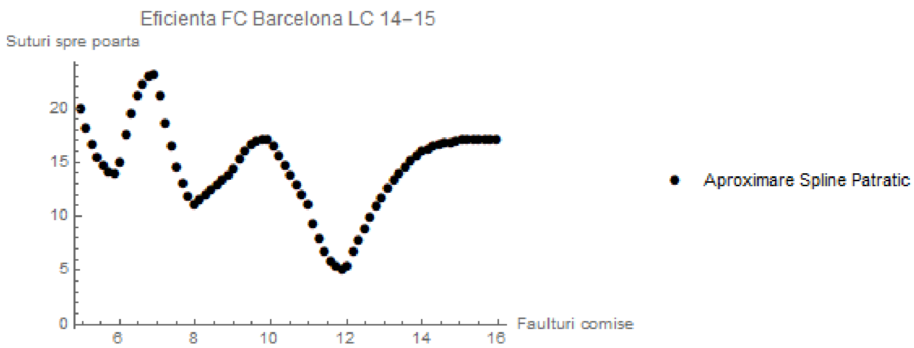
Interpolari de tip spline

Ne axam acum pe interpolarile de tip spline folosind functii de imbinare patratice si cubice. Codul Wolfram Mathematica care ne permite obtinerea acestor interpolari este urmatorul.

```
functieSplinePatraticBarcelona = Interpolation[
     suportBarcelona,
     InterpolationOrder->2][x]
]

puncteSplinePatraticBarcelona = Table[
     functieSplinePatraticBarcelona,
     {x, 5, 16, 0.15}
]

plotSplinePatraticBarcelona = ListPlot[
     puncteSplinePatraticBarcelona,
     DataRange -> {5, 16},
```

```
     AxesOrigin -> {4, 0},
     AxesLabel->{"Faulturi comise", "Suturi spre poarta"},
     PlotLabel->"Eficienta FC Barcelona LC 14-15",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare Spline Patratic"},
     PlotMarkers -> {"\[FilledCircle]"}
]
```

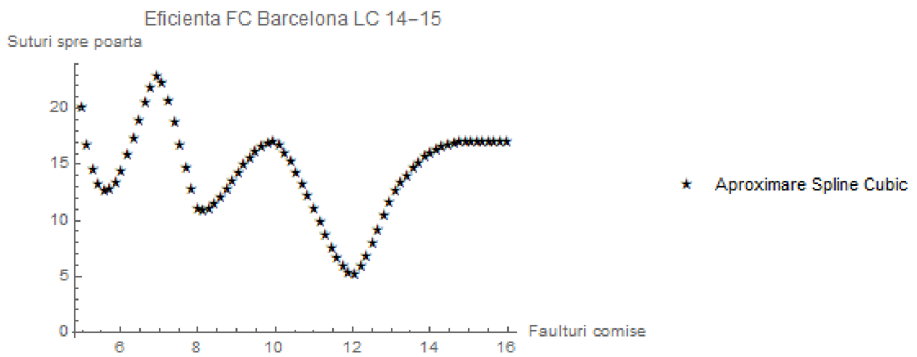
Graficul interpolarii spline cu functii patratice este cel de mai jos.



Dupa cum se poate observa, acest tip de interpolare este mult mai stabil la capete decat interpolarea Lagrange. Pentru obtinerea interpolarii spline cu functii de imbinare cubice procedeul este analog.

```
functieSplineCubicBarcelona = Interpolation[
     suportBarcelona,
     InterpolationOrder->3][x]

puncteSplineCubicBarcelona = Table[
     functieSplineCubicBarcelona,
     {x, 5, 16, 0.15}
]

plotSplineCubicBarcelona = ListPlot[
     puncteSplineCubicBarcelona,
```

```
    DataRange -> {5, 16},
    AxesOrigin -> {4, 0},
    AxesLabel->{"Faulturi comise", "Suturi spre poarta"},
    PlotLabel->"Eficienta FC Barcelona LC 14-15",
    PlotStyle->Black,
    PlotLegends -> {"Aproximare Spline Cubic"},
    PlotMarkers -> {"\[FivePointedStar]"}
]
```

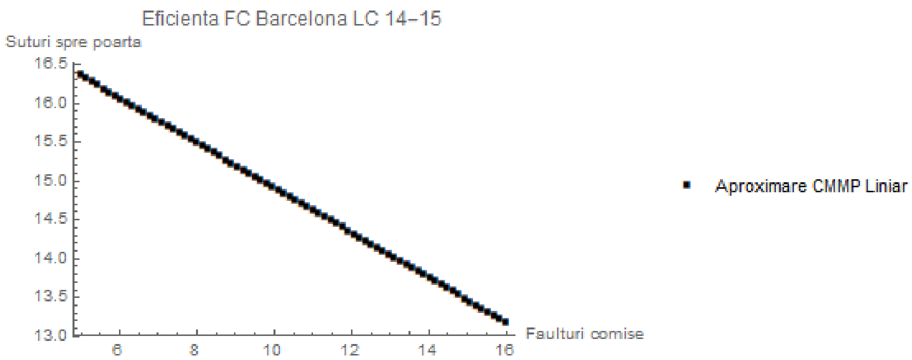


Aproximari in sensul CMMP

Ne dorim in continuare sa gasim un polinom liniar care trece printre toate punctele. Pentru a-l gasi pe cel mai bun din punct de vedere al distantei de la punctele din suportul interpolarii pana la el, vom folosi aproximarea in sensul celor mai mici patrate.

```
functieCMMPLiniarBarcelona = Fit[
    suportBarcelona,
    {1, x},
    x
]
```

```
puncteCMMPLiniarBarcelona = Table[
     functieCMMPLiniarBarcelona,
     {x, 5, 16, 0.15}
]

plotCMMPLiniarBarcelona = ListPlot[
     puncteCMMPLiniarBarcelona,
     DataRange -> {5, 16},
     AxesOrigin -> {4, 0},
     AxesLabel->{"Faulturi comise", "Suturi spre poarta"},
     PlotLabel->"Eficienta FC Barcelona LC 14-15",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare CMMP Liniar"},
     PlotMarkers -> {"\[FilledSquare]"}
]
```

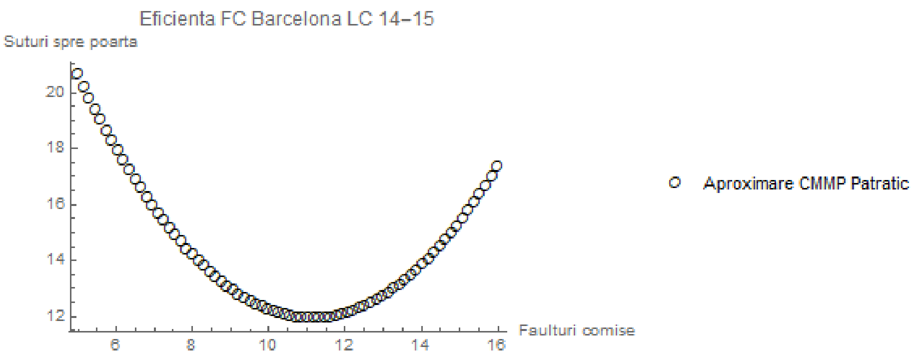


Observam ca graficul are o panta descendenta, ceea ce ne arata faptul ca numarul de suturi spre poarta a scazut proportional cu numarul de faulturi comise de-a lungul competitiei pentru FC Barcelona in sezonul 2014-2015.

Pentru a calcula aproximarea patratica in sensul celor mai mici patrate putem folosi Wolfram Mathematica astfel.

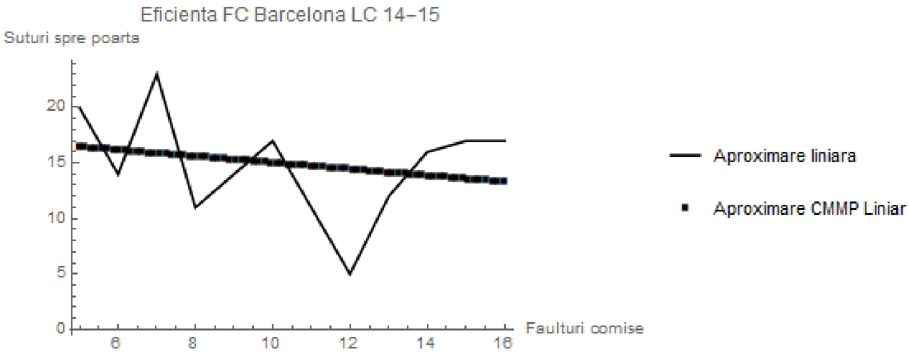
```
functieCMMPPatraticBarcelona = Fit[
     suportBarcelona,
     {1, x, x^2},
     x]

puncteCMMPPatraticBarcelona = Table[
     functieCMMPPatraticBarcelona,
     {x, 5, 16, 0.15}
]

plotCMMPPatraticBarcelona = ListPlot[
     puncteCMMPPatraticBarcelona,
     DataRange -> {5, 16},
     AxesOrigin -> {4, 0},
     AxesLabel->{"Faulturi comise", "Suturi spre poarta"},
     PlotLabel->"Eficienta FC Barcelona LC 14-15",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare CMMP Patratic"},
     PlotMarkers -> {"O"}
]
```

Trasarea graficului aproximarii patratice se face in modul urmator.

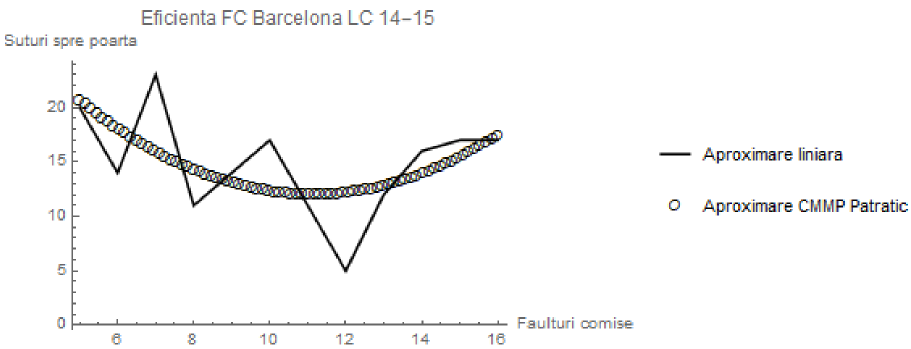


In continuare, vom afisa interpolarea liniara raportata la aproximarile in sensul celor mai mici patrate pentru a observa cum acestea trec perfect printre punctele obtinute din celelalte unsprezece meciuri.

```
Show[
     plotLiniarBarcelona,
     plotCMMPLiniarBarcelona
]
```

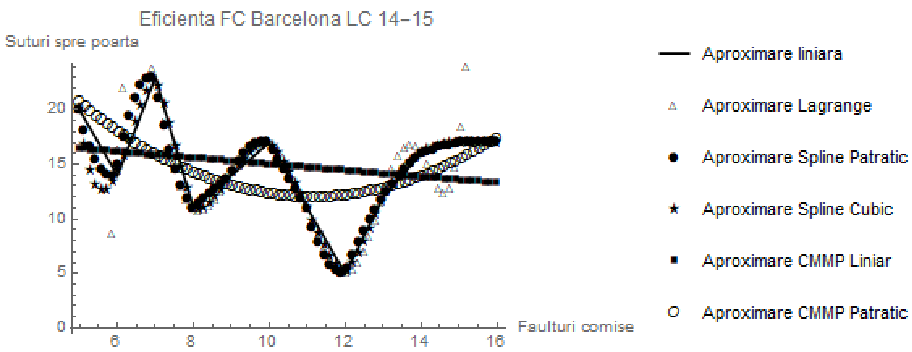


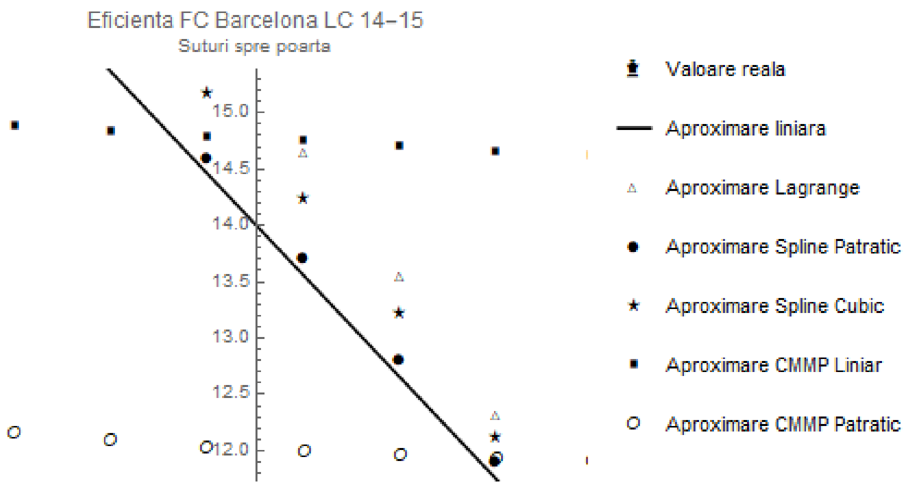

```
Show[
     plotLiniarBarcelona,
     plotCMMPLiniarBarcelona
]
```



Acum afisam toate rezultatele de la metodele folosite suprapuse pe acelasi grafic.

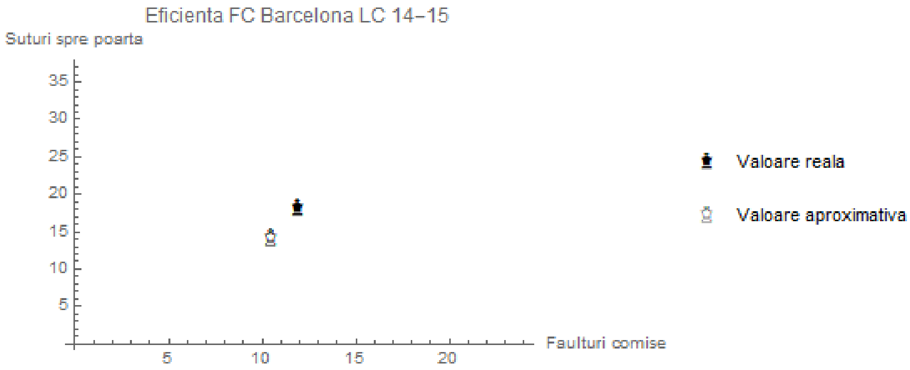
```
Show[
    plotLiniarBarcelona,
    plotLagrBarcelona,
    plotSplinePatraticBarcelona,
    plotSplineCubicBarcelona,
    plotCMMPLiniarBarcelona,
    plotCMMPPatraticBarcelona,
    PlotRange -> {{10, 11}, {11.8, 15.2}}
    AxesOrigin -> {10.5, 11}
]
```





Centrand in punctul cautat, {12, 18}, observam ca aproximarea cu un polinom patratic in sensul celor mai mici patrate nu se afla in vecinatatea valorii reale, deci putem sa nu folosim rezultatele acestei metode in aflarea estimarii dorite.

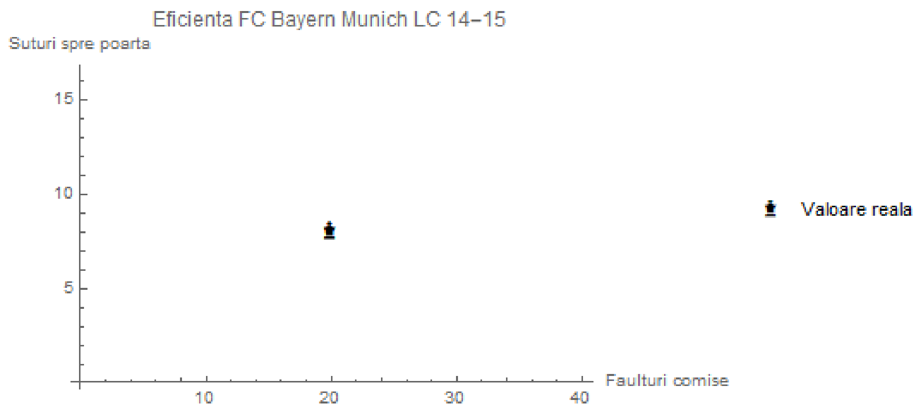
Astfel, punctul aproximat va fi {10.5, 14}, unde 10.5 este media tuturor absciselor punctelor din suportul interpolarii, iar 15 este media tuturor metodelor folosite pentru aproximarea numarului de suturi spre poarta din finala.



2.3.2 FC Bayern Munich

Acum ne propunem sa aproximam numarul de suturi spre poarta raportate la numarul de faulturi efectuate pentru echipa FC Bayern Munich. Aceasta a fost semifinalista, jucand astfel douasprezece meciuri in total in sezonul competitional 2014-2015 al Ligii Campionilor. Vom folosi datele de la toate meciurile jucate de Bayern in afara de primul meci de dupa grupe si vom incerca sa aproximam statistica dorita din acel meci, FC Shakhtar Donetsk - FC Bayern Munich 0 - 0, unde bavarezii au avut un numar de 20 de faulturi efectuate si 8 suturi spre poarta.

Valoarea reala, ({20, 8}), arata ca in graficul urmator.



Suportul interpolarii

Suportul interpolarii contine doar zece puncte in loc de unsprezece deoarece in primele doua meciuri din faza grupelor, Bayern a avut exact acelasi raport de faulturi comise si suturi spre poarta, $\{8, 22\}$.

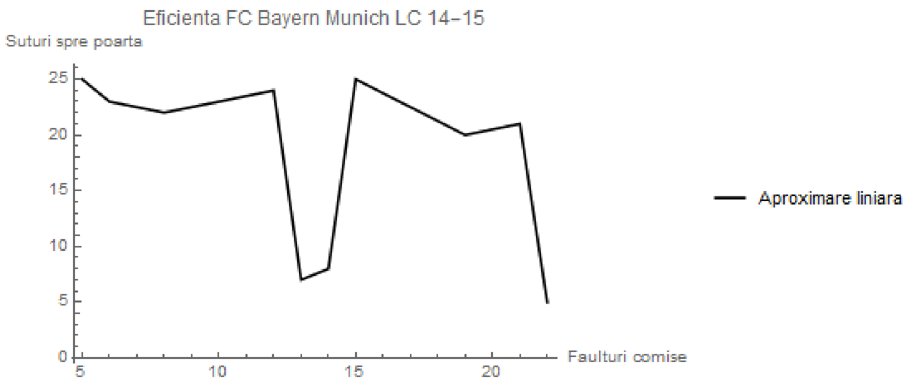
`suportBayern = {{5, 25}, {6, 23}, {8, 22}, {12, 24}, {13, 7}, {14, 8}, {15, 25}, {19, 20}, {21, 21}, {22, 5}}`

Interpolare liniara

Vom incepe cu interpolarea liniara si cum se poate genera aceasta in Wolfram Mathematica.

```
plotLiniarBayern = ListPlot[
    suportBayern,
    AxesOrigin->{4, 0},
    AxesLabel->{"Faulturi comise", "Suturi spre poarta"},
    PlotLabel->"Eficienta FC Bayern Munich LC 14-15",
    Joined->True,
    PlotStyle->Black,
    PlotLegends->{"Aproximare liniara"}
]
```

Graficul interpolarii liniare este urmatorul.



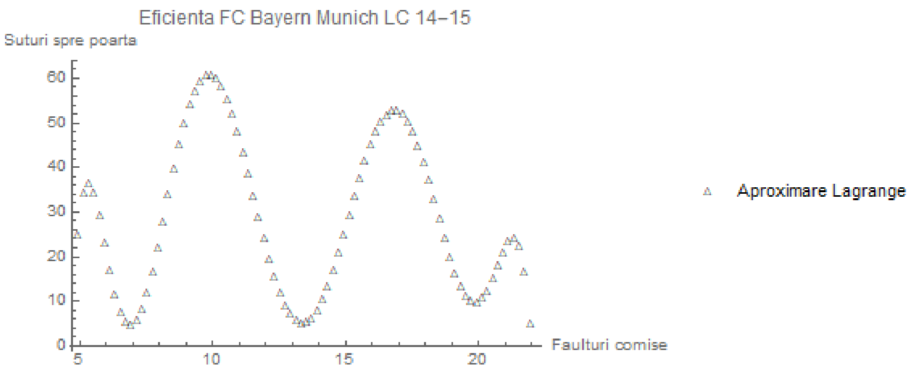
Interpolare Lagrange

Urmeaza interpolarea cu polinom Lagrange care va avea grad maxim 9, deoarece sunt 10 puncte in suportul interpolarii. Codul Wolfram Mathematica pentru generare este urmatorul.

```
functieLagrangeBayern = InterpolatingPolynomial[
     suportBayern,
     x
]

puncteLagrangeBayern = Table[
     functieLagrangeBayern,
     {x, 5, 22, 0.2}
]

plotLagrangeBayern = ListPlot[
     puncteLagrangeBayern,
     DataRange -> {5, 22},
     AxesOrigin -> 4, 0,
     AxesLabel->{"Faulturi comise", "Suturi spre poarta"},
     PlotLabel->"Eficienta FC Bayern Munich LC 14-15",
```

```
    PlotStyle->Black,
    PlotLegends -> {"Aproximare Lagrange"},
    PlotMarkers -> {"\[EmptyUpTriangle]"}
]
```

Mai jos este prezentat graficul interpolarii Lagrange pentru FC Bayern Munich in anul competitional 2014-2015. Putem observa ca de data aceasta nu exista valori negative pe grafic sau oscilatii foarte mari, ceea ce inseamna ca modul in care se comporta polinomul Lagrange depinde considerabil de felul in care arata punctele.



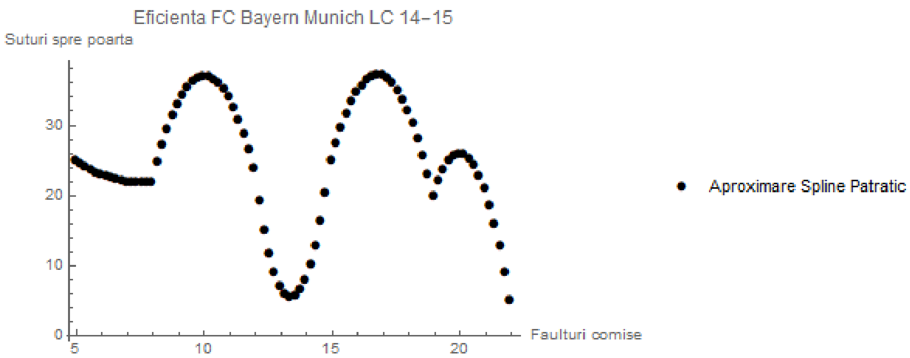
Interpolari de tip spline

In continuare, vom folosi interpolari de tip spline cu functii de imbinare polinoame de grad 2 si 3. Pentru polinoame patratice, procedeul este urmatorul.

```
functieSplinePatraticBayern = Interpolation[
    suportBayern,
    InterpolationOrder->2][x]
]

puncteSplinePatraticBayern = Table[
    functieSplinePatraticBayern,
```

```
    {x, 5, 22, 0.2}
]

plotSplinePatraticBayern = ListPlot[
    puncteSplinePatraticBayern,
    DataRange -> {5, 22},
    AxesOrigin -> {4, 0},
    AxesLabel->{"Faulturi comise", "Suturi spre poarta"},
    PlotLabel->"Eficienta FC Bayern Munich LC 14-15",
    PlotStyle->Black,
    PlotLegends -> {"Aproximare Spline Patratic"},
    PlotMarkers -> {"\[FilledCircle]"}
]
```

Mai jos se poate observa graficul care reprezinta interpolarea spline cu functii patratice de imbinare.

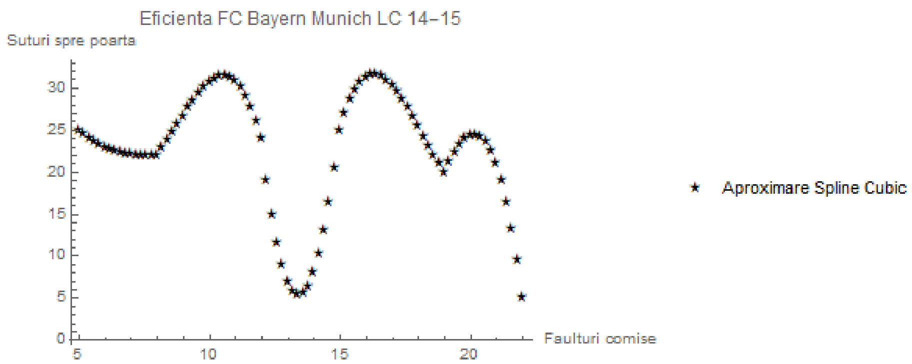


Pentru abordarea cu functii cubice de imbinare, algoritmul este la fel ca la cele patratice.

```
functieSplineCubicBayern = Interpolation[
    suportBayern,
    InterpolationOrder->3][x]
]
```

```
puncteSplineCubicBayern = Table[
     functieSplineCubicBayern,
     {x, 5, 22, 0.2}
]

plotSplineCubicBayern = ListPlot[
     puncteSplineCubicBayern,
     DataRange -> {5, 22},
     AxesOrigin -> {4, 0},
     AxesLabel->{"Faulturi comise", "Suturi spre poarta"},
     PlotLabel->"Eficienta FC Bayern Munich LC 14-15",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare Spline Cubic"},
     PlotMarkers -> {"\[FivePointedStar]"}
]
```

Graficul interpolarii spline cu functii de imbinare cubice este urmatorul.

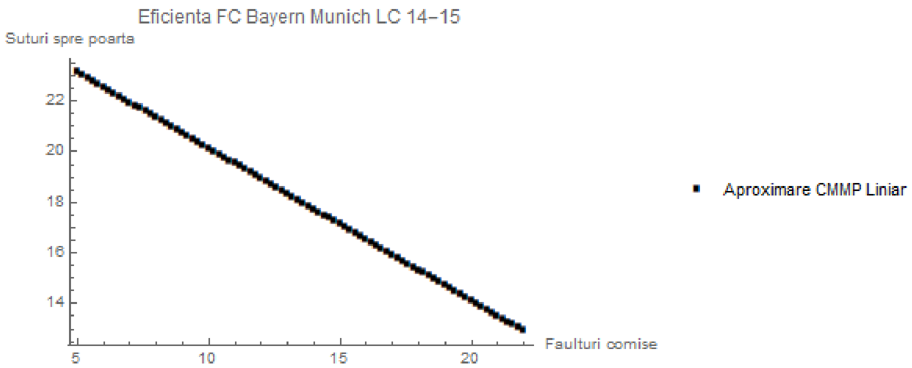


Aproximari in sensul CMMP

Urmeaza aproximarile in sensul celor mai mici patrate. Vom incepe, dupa cum ne-am obisnuit, cu aproximarea liniara si cu modul in care aceasta se genereaza in Wolfram Mathematica. Apoi vom continua cu aproximarea patratica si cu comparatii ale acestor doua aproximari cu interpolarea liniara.

```
functieCMMPLiniarBayern = Fit[suportBayern, {1, x}, x]

puncteCMMPLiniarBayern = Table[
     functieCMMPLiniarBayern,
     {x, 5, 22, 0.2}
]

plotCMMPLiniarBayern = ListPlot[
     puncteCMMPLiniarBayern,
     DataRange -> {5, 22},
     AxesOrigin -> {4, 0},
     AxesLabel->{"Faulturi comise", "Suturi spre poarta"},
     PlotLabel->"Eficienta FC Bayern Munich LC 14-15",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare CMMP Liniar"},
     PlotMarkers -> {"\[FilledSquare]"}
]
```

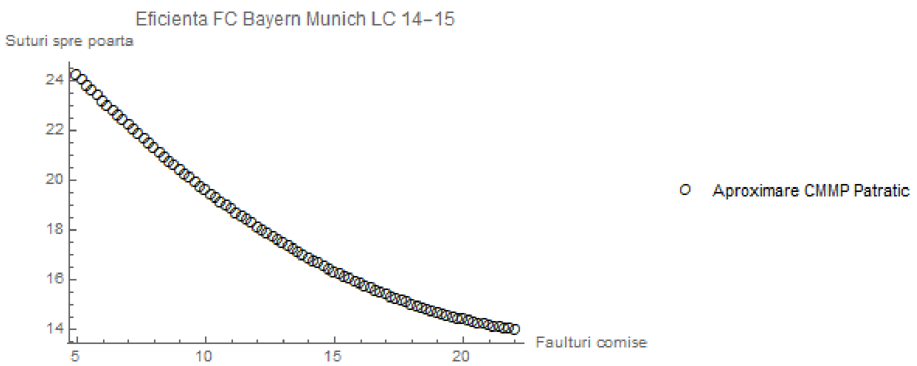
In graficul urmator se poate observa o coborare liniara care inseamna ca numarul de faulturi comise este invers proportional cu numarul de suturi spre poarta. O echipa care faulteaza mult va trage mai putin spre poarta.



Aproximarea patratica in sensul celor mai mici patrate se face aproape identic cu cea liniara.

```
functieCMMPPatraticBayern = Fit[suportBayern, {1, x, x^2}, x]

puncteCMMPPatraticBayern = Table[
     functieCMMPPatraticBayern,
     {x, 5, 22, 0.2}
]

plotCMMPPatraticBayern = ListPlot[
     puncteCMMPPatraticBayern,
     DataRange -> {5, 22},
     AxesOrigin -> {4, 0},
     AxesLabel->{"Faulturi comise", "Suturi spre poarta"},
     PlotLabel->"Eficienta FC Bayern Munich LC 14-15",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare CMMP Patratic"},
     PlotMarkers -> {"O"}
]
```

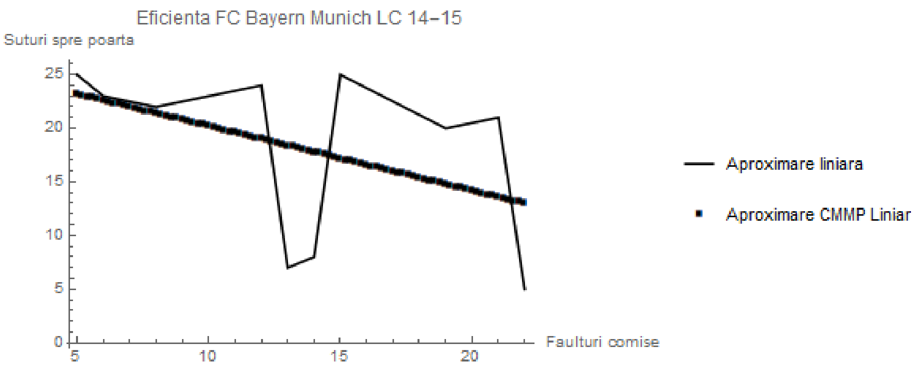
Graficul aproximarii in sensul celor mai mici patrate printr-ul polinom de gradul al doilea arata ca in figura de mai jos.



Acum vom arata diferenta dintre aproximarea liniara in sensul celor mai mici patrate care nu trece prin ci printre punctele din suportul interpolarii si interpolarea liniara care va trece exact prin acele puncte.

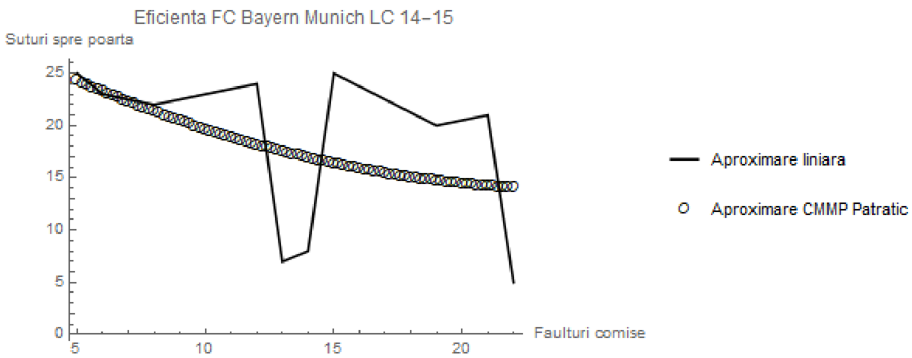
Deoarece cele doua aproximari sunt deja calculate, le putem folosi plotarile pentru a le suprapune.

```
Show[plotLiniarBayern, plotCMMPLiniarBayern]
```



Analog, observam cum punctele din aproximarea patratica in sensul celor mai mici patrate trec exact printre punctele din suportul interpolarii.

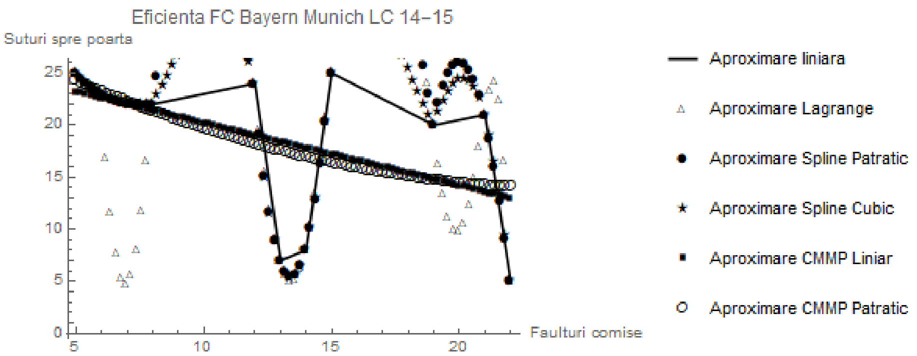
```
Show[plotLiniarBayern, plotCMMPLiniarBayern]
```

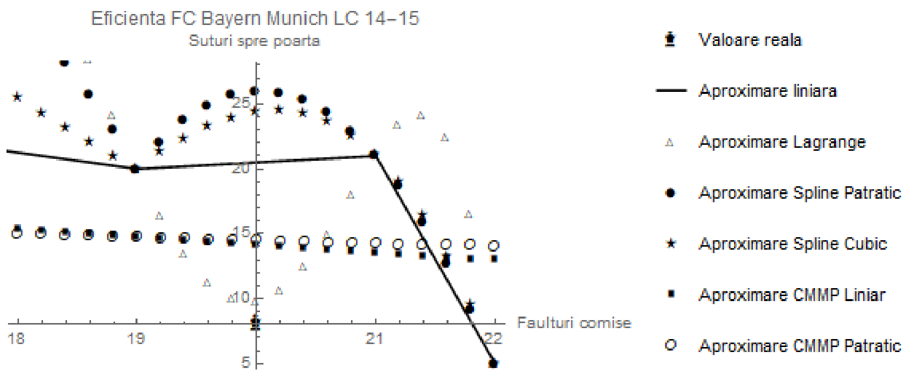


Dupa ce am construit toate interpolarile si aproximarile propuse urmeaza sa le afisam in acelasi grafic, iar apoi sa il centram in punctul reprezentand valoarea reala pe care o cautam ($\{20, 8\}$), adica numarul de faulturi comise si numarul de suturi la poarta din primul meci de dupa faza grupelor, pentru a observa mai usor diferentele si eroarea fiecarei metode aplicate.

Folosim toate variabilele create pentru plotare si doar setam marginile graficului, pe ambele axe, Wolfram Mathematica ajutandu-ne cu o sintaxa facila in acest sens.

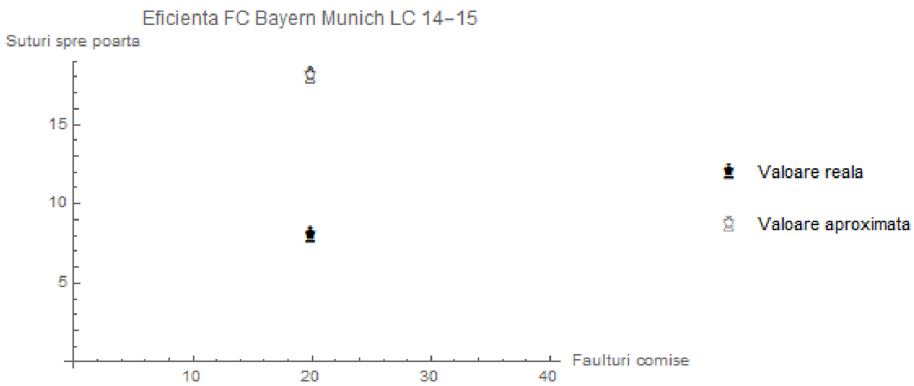
```
Show[
     plotLiniarBayern,
     plotLagrBayern,
     plotSplinePatraticBayern,
     plotSplineCubicBayern,
     plotCMMPLiniarBayern,
     plotCMMPPatraticBayern,
     PlotRange -> {{18, 22}, {5, 27}}
     AxesOrigin -> {20, 8}
]
```





Vom folosi toate rezulatele obtinute si, astfel, in punctul de coordonata $x = 20$ vom aproxima $y = 18$.

Observam ca distanta dintre valoarea reala si valoarea aproximata este egala cu 10, ceea ce este mult mai mult decat ne-am dori.

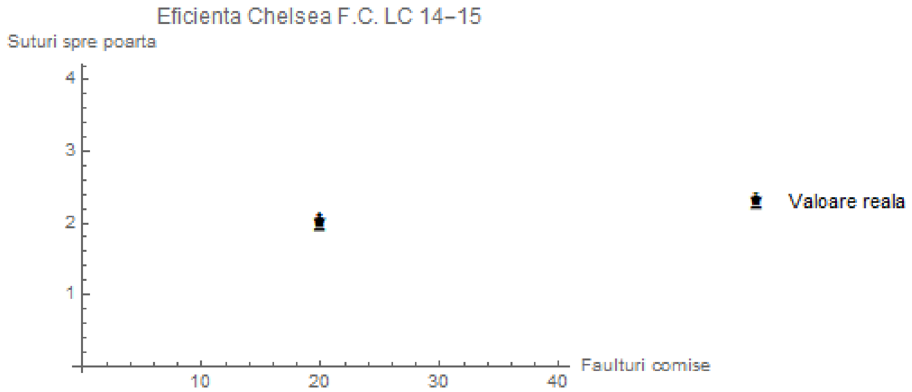


2.3.3 Chelsea F.C.

Datorita faptului ca Chelsea F.C. a jucat 8 meciuri in sezonul competitional 2014-2015 al Ligii Campionilor, fiind eliminata in optimile competitiei de Paris Saint-Germain F.C., echipa care urma sa fie eliminata in sferturi de FC Barcelona, ne dorim sa aproximam numarul de suturi spre poarta din primul meci de dupa faza grupelor, Paris Saint-Germain

F.C. - Chelsea F.C. 1 - 1. In acest meci, britanicii au comis 20 de faulturi si 2 suturi spre poarta.

Graficul care evidentiaza valoarea reala pe care dorim sa o aproximam este urmatorul.



Suportul interpolarii

Avem sapte puncte in suportul interpolarii, incercand sa aproximam al optulea rezultat cunoscut.

```
suportChelsea = {{7, 23}, {9, 13}, {10, 15}, {11, 17}, {13,
15}, {16, 19}, {24, 14}}
```

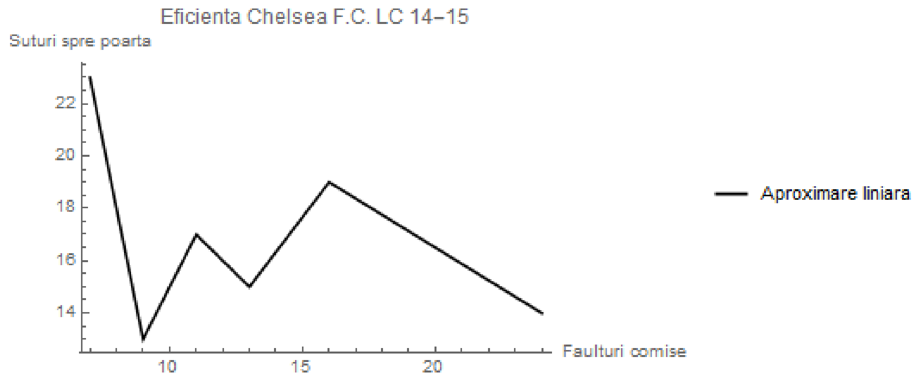
Interpolare liniara

Incepem cu interpolarea liniara si observam ca procedeul de generare a acesteia in Wolfram Mathematica este similar cu cel din subcapitolele trecute.

```
plotLiniarChelsea = ListPlot[
     suportChelsea,
     AxesOrigin->{6, 0},
     AxesLabel->{"Faulturi comise", "Suturi spre poarta"},
     PlotLabel->"Eficienta Chelsea F.C. LC 14-15",
```

```
    Joined->True,
    PlotStyle->Black,
    PlotLegends->{"Aproximare liniara"}
]
```

Graficul acestei interpolari ne duce cu gandul la un polinom initial convex, care isi schimba convexitatea la mijlocul suportului interpolarii.



Interpolare Lagrange

Pentru interpolarea Lagrange generam punctele pe care le vom afisa pe grafic astfel.

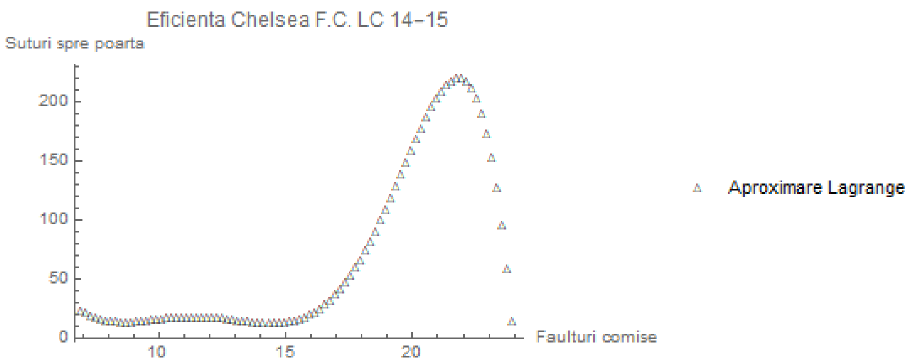
```
functieLagrangeChelsea = InterpolatingPolynomial[
    suportChelsea,
    x
]
```

Setam ca margini 7 si 24, adica cel mai mic numar de faulturi comise intr-un meci si, respectiv, cel mai mare. Distanta 0.2 dintre puncte este pentru o mai buna vizualizare a graficului, fiindca daca am fi ales ceva mai mare nu s-ar fi generate destule puncte.

```
puncteLagrangeChelsea = Table[
```

```
    functieLagrangeChelsea,
    {x, 7, 24, 0.2}
]

  plotLagrangeChelsea = ListPlot[
    puncteLagrangeChelsea,
    DataRange -> {7, 24},
    AxesOrigin -> 6, 0,
    AxesLabel->{"Faulturi comise", "Suturi spre poarta"},
    PlotLabel->"Eficienta Chelsea F.C. LC 14-15",
    PlotStyle->Black,
    PlotLegends -> {"Aproximare Lagrange"},
    PlotMarkers -> {"\[EmptyUpTriangle]"}
]
```

Graficul interpolarii Lagrange este urmatorul.



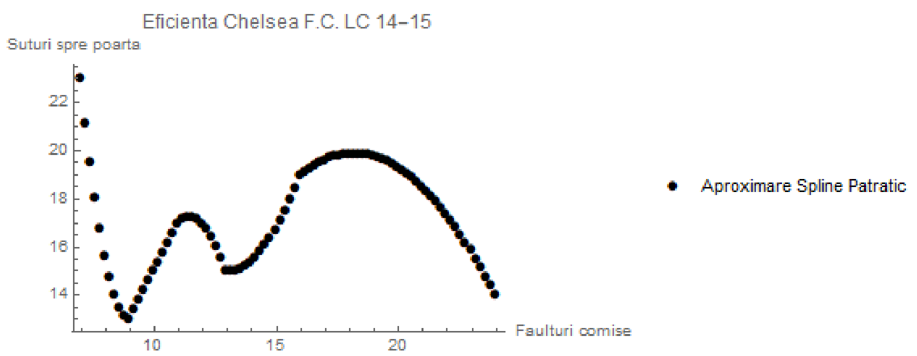
Interpolari de tip spline

Urmeaza interpolarile de tip spline care folosesc functii de imbinare patratice si cubice. Pentru generarea acestora in Wolfram Mathematica se efectueaza urmatoarele operatii.

```
functieSplinePatraticChelsea = Interpolation[
```

```
    suportChelsea,
    InterpolationOrder->2][x]
]

puncteSplinePatraticChelsea = Table[
    functieSplinePatraticChelsea,
    {x, 7, 24, 0.2}
]

plotSplinePatraticChelsea = ListPlot[
    puncteSplinePatraticChelsea,
    DataRange -> {7, 24},
    AxesOrigin -> {6, 0},
    AxesLabel->{"Faulturi comise", "Suturi spre poarta"},
    PlotLabel->"Eficienta Chelsea F.C. LC 14-15",
    PlotStyle->Black,
    PlotLegends -> {"Aproximare Spline Patratic"},
    PlotMarkers -> {"\[FilledCircle]"}
]
```

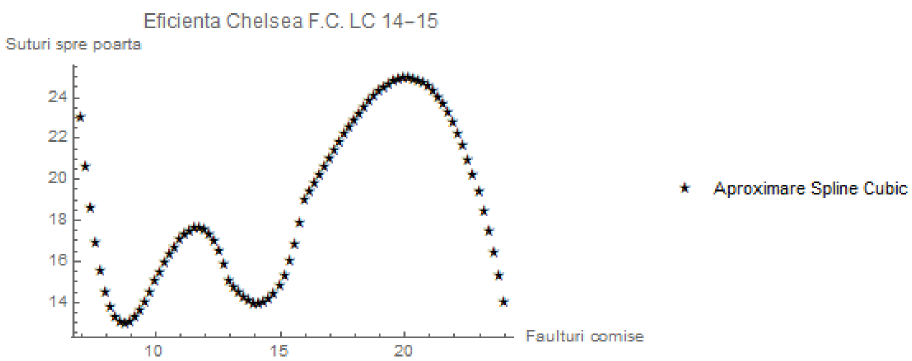
Graficul interpolarii spline realizate cu ajutorul functiilor patratice de imbinare arata astfel.



Ne ocupam in continuare de generarea interpolarii spline cu ajutorul functiilor cubice de imbinare.

```
functieSplineCubicChelsea = Interpolation[
     suportChelsea,
     InterpolationOrder->3][x]
]

puncteSplineCubicChelsea = Table[
     functieSplineCubicChelsea,
     {x, 7, 24, 0.2}
]

plotSplineCubicChelsea = ListPlot[
     puncteSplineCubicChelsea,
     DataRange -> {7, 24},
     AxesOrigin -> {6, 0},
     AxesLabel->{"Faulturi comise", "Suturi spre poarta"},
     PlotLabel->"Eficienta Chelsea F.C. LC 14-15",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare Spline Cubic"},
     PlotMarkers -> {"\[FivePointedStar]"}
]
```

Graficul reprezentativ interpolarii cu functii cubice de imbinare este urmatorul.

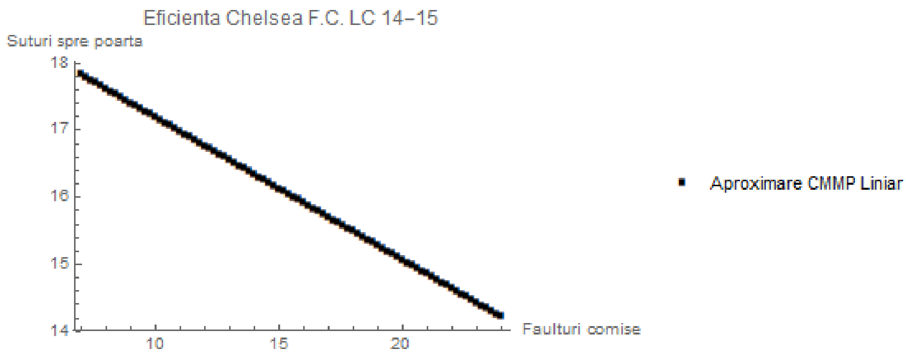


Aproximari in sensul CMMP

Dorim sa gasim o succesiune de puncte care trec printre toate punctele suportului interpolarii. Pentru inceput, vom incerca sa gasim o dreapta.

```
functieCMMPLiniarChelsea = Fit[suportChelsea, {1, x}, x]

puncteCMMPLiniarChelsea = Table[
     functieCMMPLiniarChelsea,
     {x, 7, 24, 0.2}
]

plotCMMPLiniarChelsea = ListPlot[
     puncteCMMPLiniarChelsea,
     DataRange -> {7, 24},
     AxesOrigin -> {6, 0},
     AxesLabel->{"Faulturi comise", "Suturi spre poarta"},
     PlotLabel->"Eficienta Chelsea F.C. LC 14-15",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare CMMP Liniar"},
     PlotMarkers -> {"\[FilledSquare]"}
]
```

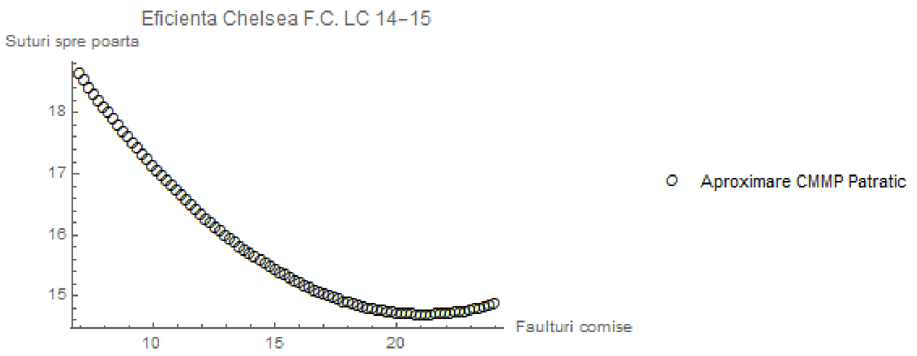
Aproximarea liniara in sensul celor mai mici patrate arata ca mai jos. Observam ca aceasta coboara.



Cautam si un polinom patratic care sa treaca printre toate punctele suportului interpolarii.

```
functieCMMPPatraticChelsea = Fit[suportChelsea, {1, x, x^2}, x]

puncteCMMPPatraticChelsea = Table[
     functieCMMPPatraticChelsea,
     {x, 7, 24, 0.2}
]

plotCMMPPatraticChelsea = ListPlot[
     puncteCMMPPatraticChelsea,
     DataRange -> {7, 24},
     AxesOrigin -> {6, 0},
     AxesLabel->{"Faulturi comise", "Suturi spre poarta"},
     PlotLabel->"Eficienta Chelsea F.C. LC 14-15",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare CMMP Patratic"},
     PlotMarkers -> {"O"}
]
```

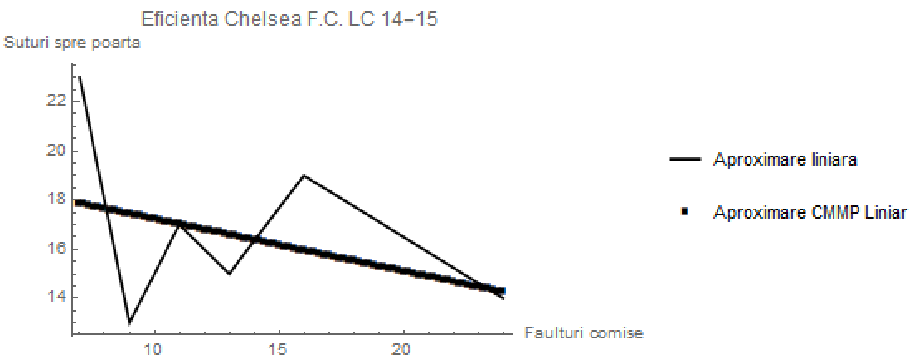
Graficul aproximarii patratice in sensul celor mai mici patrate este urmatorul si putem observa ca este convex.



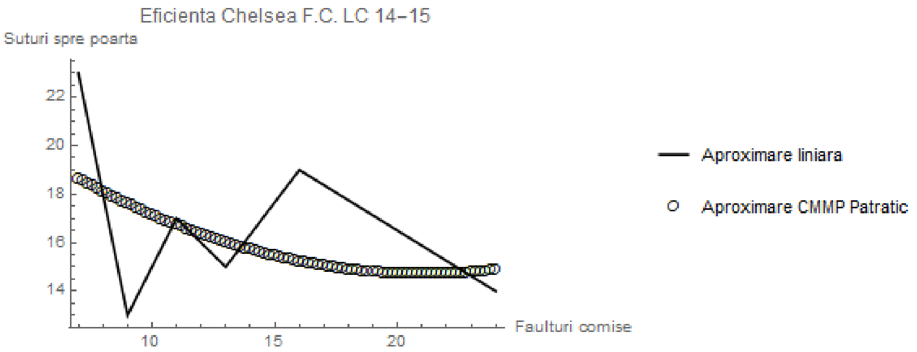
Urmeaza sa suprapunem graficele interpolarii liniare si aproximarilor in sensul celor mai mici patrate pentru a vedea la o scara mai mare daca se valideaza proprietatea de a trece printre toate punctele.

Avand deja plotarile efectuate ne mai ramane doar sa le afisam impreuna folosind functia ”Show” din Wolfram Mathematica.

```
Show[plotLiniarChelsea, plotCMMPLiniarChelsea]
```

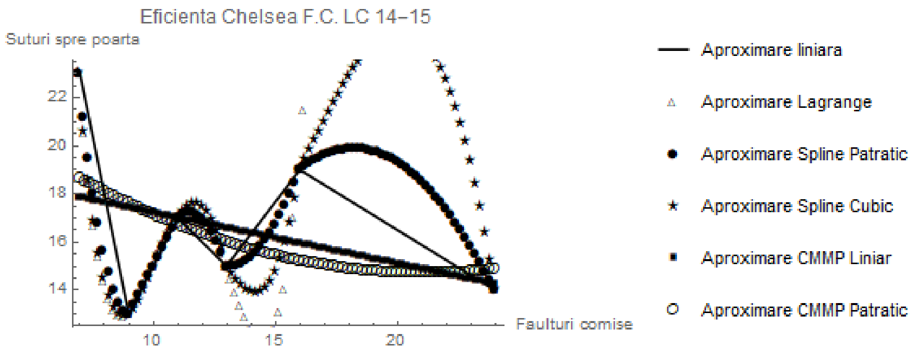



```
Show[plotLiniarChelsea, plotCMMPLiniarChelsea]
```

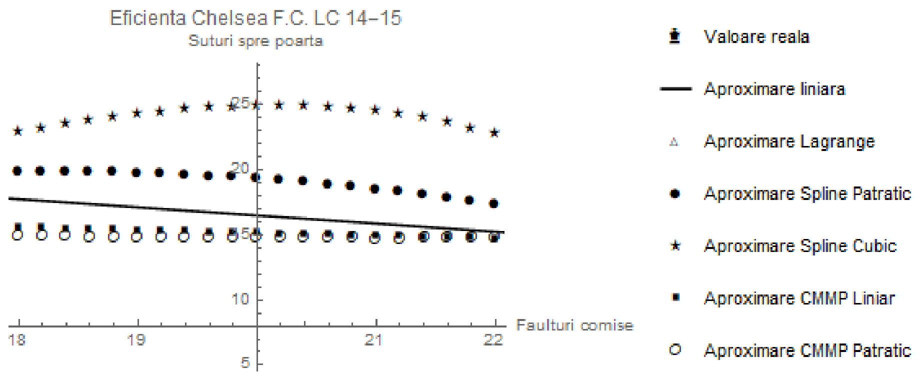


In continuare afisam toate aproximarile calculate si apoi centram in punctul $\{20, 2\}$ care reprezinta valoarea reala pe care o cautam.

```
Show[
    plotLiniarChelsea,
    plotLagrChelsea,
    plotSplinePatraticChelsea,
    plotSplineCubicChelsea,
    plotCMMPLiniarChelsea,
    plotCMMPPatraticChelsea,
    PlotRange -> {{18, 22}, {5, 27}}
    AxesOrigin -> {20, 8}
]
```

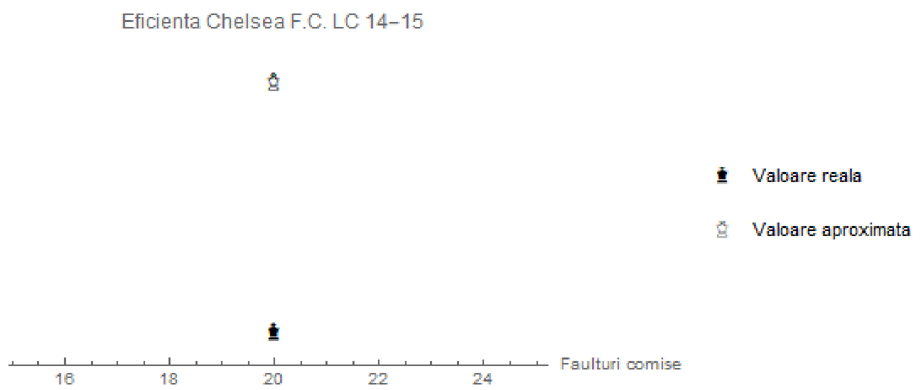


Observam in graficul centrat de mai jos ca toate aproximarile, in afara de cea prin interpolare Lagrange, pot fi folosite pentru estimarea dorita.



Gasim astfel o aproximare in punctul $\{20, 18\}$ care este la diferenta 16 de valoarea reala.

Valoarea reala si valoarea aproximativa arata grafic astfel.



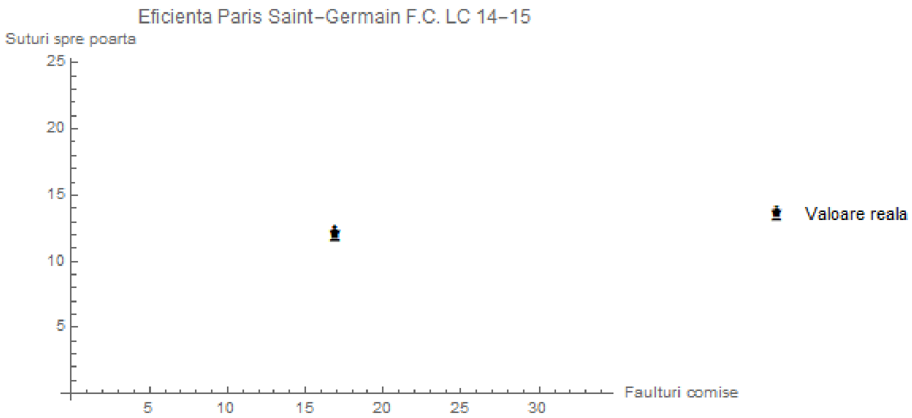
2.3.4 Paris Saint-Germain F.C.

Ne propunem sa aproximam numarul de suturi spre poarta ale echipei Paris Saint-Germain F.C. raportat la numarul de faulturi comise in al doilea meci jucat dupa faza grupelor: Chelsea FC - Paris Saint-Germain

F.C., care s-a incheiat cu scorul de 2-2.

In acest sezon competitional, 2014-2015, Paris Saint-Germain F.C. a jucat zece meciuri in total, acestia oprindu-se in faza sferturilor de finala.

In graficul de mai jos observam valoarea reala din meciul pe care dorim sa il aproximam, si anume 12 suturi spre poarta avand un numar de 17 faulturi comise.



Suportul interpolarii

Primul pas in realizarea aproximarilor este strangerea datelor din celelalte meciuri pe care Paris Saint-Germain F.C. le-a jucat in afara celui de-al doilea meci jucat dupa faza grupelor si introducerea lor intr-un vector de puncte numit "suportul interpolarii".

Observam ca accest vector este ordonat dupa primul termen, cel care reprezinta numarul de faulturi comise, deoarece acest lucru este necesar pentru a putea aplica metodele enumerate in capitolul 1.

```
suportPSG = {{7, 12}, {8, 13}, {9, 14}, {10, 8}, {11, 18}, {12,
10}, {15, 10}, {16, 8}, {20, 14}}
```

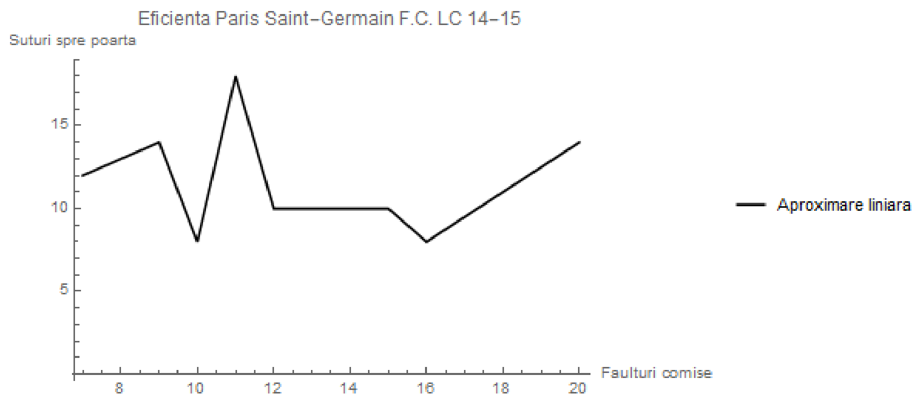
Interpolare liniara

Interpolarea liniara este prima metoda numerica pe care o vom aborda pentru o reprezentare grafica a suportului interpolarii. Aceasta reprezinta trasarea unor segmente intre fiecare doua puncte consecutive.

Putem observa codul Wolfram Mathematica folosit pentru a genera graficul interpolarii liniare mai jos.

```
plotLiniarPSG = ListPlot[
     suportPSG,
     AxesOrigin->{6, 0},
     AxesLabel->{"Faulturi comise", "Suturi spre poarta"},
     PlotLabel->"Eficienta Paris Saint-Germain F.C. LC 14-15",
     Joined->True,
     PlotStyle->Black,
     PlotLegends->{"Aproximare liniara"}
]
```

Graficul interpolarii liniare arata in modul urmator.



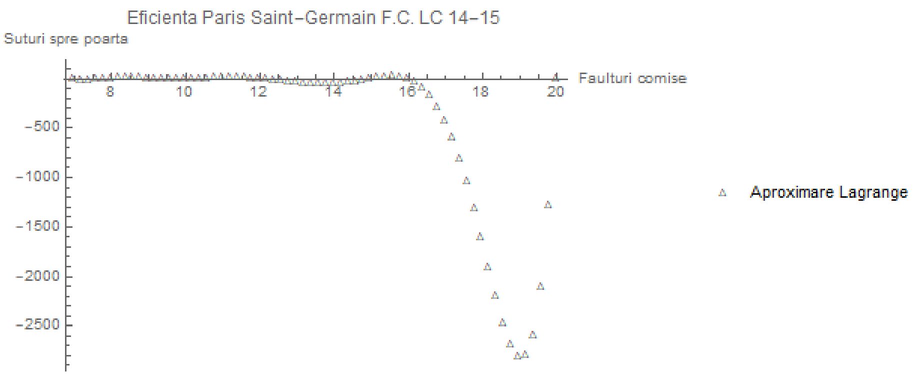
Interpolare Lagrange

Urmeaza interpolarea cu ajutorul polinomului Lagrange care va avea grad maxim 8 deoarece folosim 9 puncte in suportul interpolarii.

Abordarea in Wolfram Mathematica este urmatoarea.

```
functieLagrangePSG = InterpolatingPolynomial[
     suportPSG,
     x
]

puncteLagrangePSG = Table[
     functieLagrangePSG,
     {x, 7, 20, 0.2}
]

plotLagrangePSG = ListPlot[
     puncteLagrangePSG,
     DataRange -> {7, 20},
     AxesOrigin -> 6, 0,
     AxesLabel->{"Faulturi comise", "Suturi spre poarta"},
     PlotLabel->"Eficienta Paris Saint-Germain F.C. LC 14-15",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare Lagrange"},
     PlotMarkers -> {"\[EmptyUpTriangle]"}
]
```

Graficul interpolarii Lagrange este urmatorul.

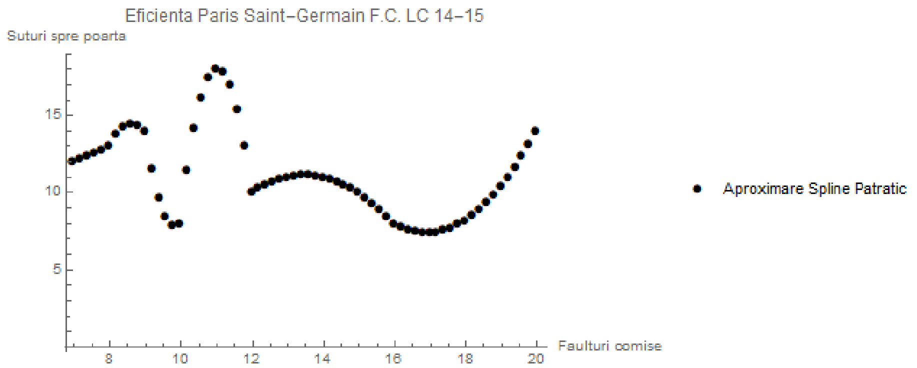


Interpolari de tip spline

Urmeaza interpolarile cu functii de imbinare de tip spline. Vom incepe cu functii spline patratice si cu modul in care se pot genera aceaste interpolari in Wolfram Mathematica.

```
functieSplinePatraticPSG = Interpolation[
     suportPSG,
     InterpolationOrder->2][x]
]
puncteSplinePatraticPSG = Table[
     functieSplinePatraticPSG,
     {x, 7, 20, 0.2}
]

plotSplinePatraticPSG = ListPlot[
     puncteSplinePatraticPSG,
     DataRange -> {7, 20},
     AxesOrigin -> {6, 0},
     AxesLabel->{"Faulturi comise", "Suturi spre poarta"},
     PlotLabel->"Eficienta Paris Saint-Germain F.C. LC 14-15",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare Spline Patratic"},
     PlotMarkers -> {"\[FilledCircle]"}
]
```

Graficul interpolarii spline cu ajutorul functiilor de imbinare patratice arata astfel.

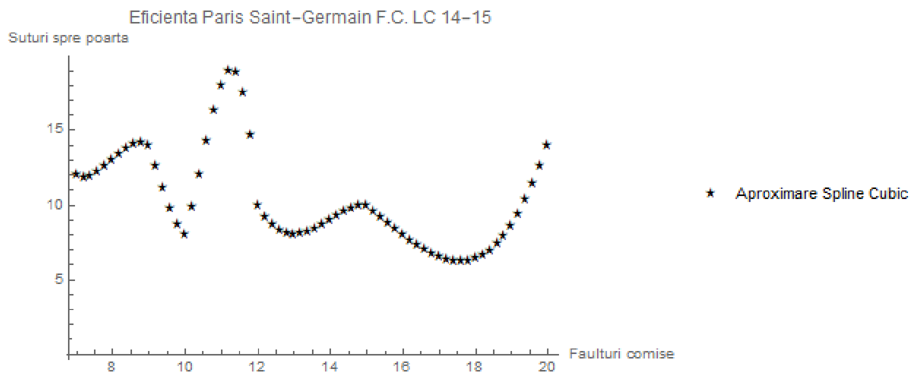


Observam ca spre deosebire de interpolarea cu ajutorul polinomului Lagrange, valorile din capetele suportului de interpolare sunt mult mai apropiate de niste valori posibil reale (7 de suturi spre poarta in loc de -2000). Pentru spline-uri cu functii cubice, procedeul este asemanator cu cel folosit pentru functiile patratice.

```
functieSplineCubicPSG = Interpolation[
     suportPSG,
     InterpolationOrder->3][x]
]

puncteSplineCubicPSG = Table[
     functieSplineCubicPSG,
     {x, 7, 20, 0.2}
]

plotSplineCubicPSG = ListPlot[
     puncteSplineCubicPSG,
     DataRange -> {7, 20},
     AxesOrigin -> {6, 0},
     AxesLabel->{"Faulturi comise", "Suturi spre poarta"},
     PlotLabel->"Eficienta Paris Saint-Germain F.C. LC 14-15",
     PlotStyle->Black,
```

```
    PlotLegends -> {"Aproximare Spline Cubic"},
    PlotMarkers -> {"\[FivePointedStar]"}
]
```

Graficul interpolarii spline cu functii de imbinare cubice este urmatorul.



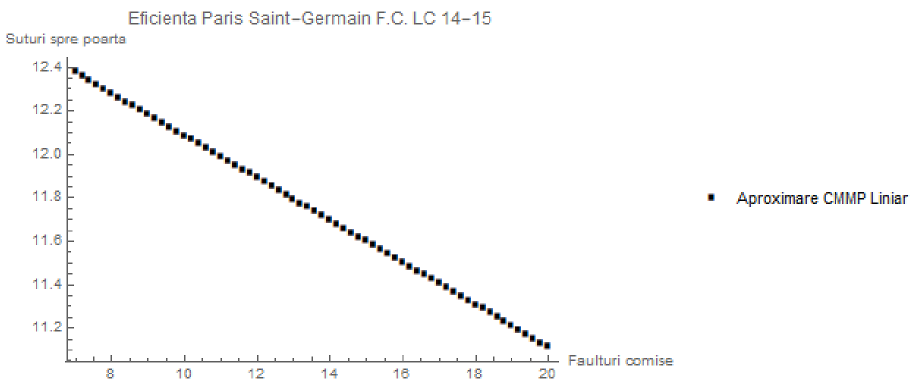
Aproximari in sensul CMMP

In continuare vom aproxima o functie liniara care trece printre punctele din suportul interpolarii in sensul celor mai mici patrate.

```
functieCMMPLiniarPSG = Fit[suportPSG, {1, x}, x]

puncteCMMPLiniarPSG = Table[
    functieCMMPLiniarPSG,
    {x, 7, 20, 0.2}
]

plotCMMPLiniarPSG = ListPlot[
    puncteCMMPLiniarPSG,
    DataRange -> {7, 20},
    AxesOrigin -> {6, 0},
```

```
     AxesLabel->{"Faulturi comise", "Suturi spre poarta"},
     PlotLabel->"Eficienta Paris Saint-Germain F.C. LC 14-15",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare CMMP Liniar"},
     PlotMarkers -> {"\[FilledSquare]"}
]
```

Graficul aproximarii liniare in sensul celor mai mici patrate arata in felul urmator si vom compara putin mai tarziu cu interpolarea liniara ca sa observam pozitia relativa a punctelor aproximate fata de cele din suportul interpolarii.

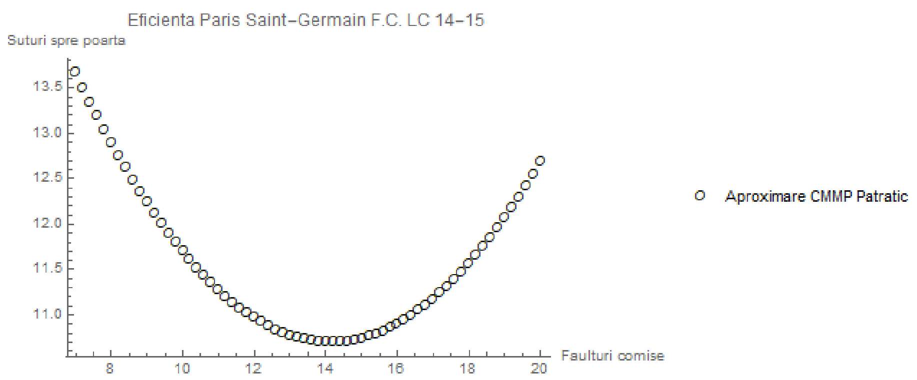


De data aceasta, vom alege un polinom patratic care sa treaca printre toate punctele pentru o eventuala aproximare mai buna.

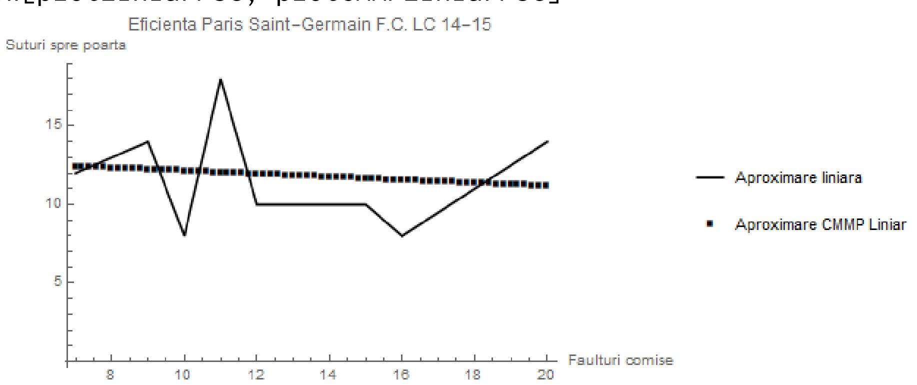
```
functieCMMPPatraticPSG = Fit[suportPSG, {1, x, x^2}, x]

puncteCMMPPatraticPSG = Table[
     functieCMMPPatraticPSG,
     {x, 7, 20, 0.2}
]

plotCMMPPatraticPSG = ListPlot[
     puncteCMMPPatraticPSG,
```

```
    DataRange -> {7, 20},
    AxesOrigin -> {6, 0},
    AxesLabel->{"Faulturi comise", "Suturi spre poarta"},
    PlotLabel->"Eficienta Paris Saint-Germain F.C. LC 14-15",
    PlotStyle->Black,
    PlotLegends -> {"Aproximare CMMP Patratic"},
    PlotMarkers -> {"O"}
]
```

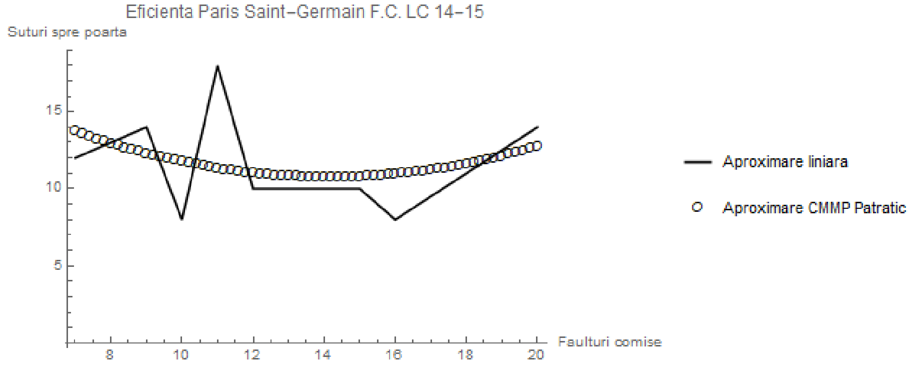
Graficul aproximarii patratice in sensul celor mai mici patrate este urmatorul.



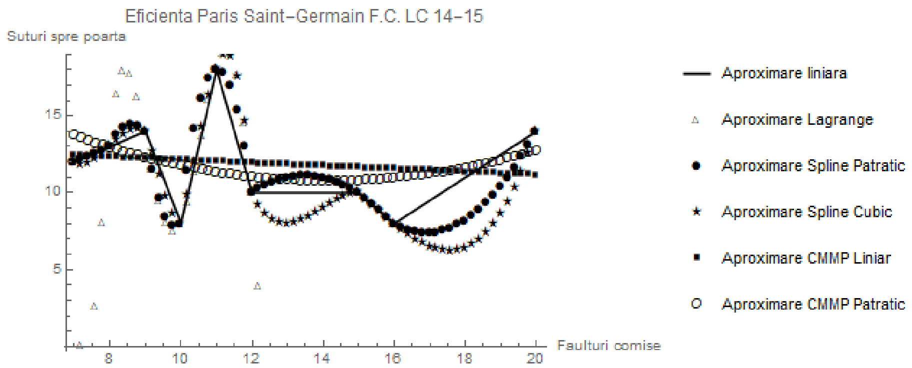

```
Show[plotLiniarPSG, plotCMMPLiniarPSG]
```



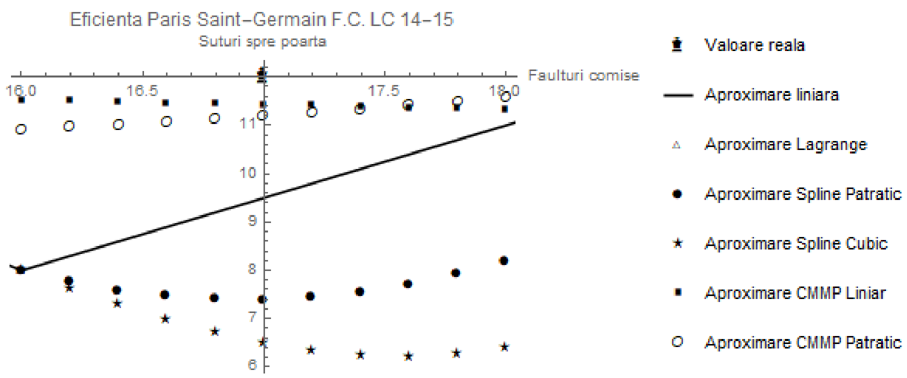
```
Show[plotLiniarPSG, plotCMMPLiniarPSG]
```



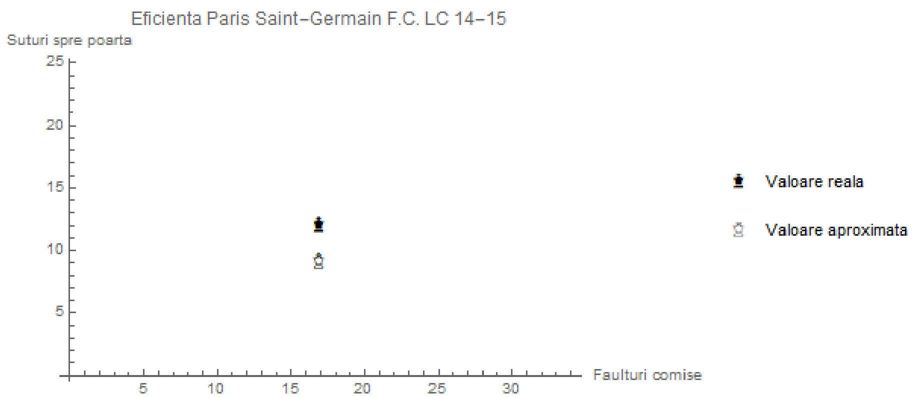

```
Show[
    plotLiniarPSG,
    plotLagrPSG,
    plotSplinePatraticPSG,
    plotSplineCubicPSG,
    plotCMMPLiniarPSG,
    plotCMMPPatraticPSG,
    PlotRange -> {{16, 18}, {6, 12}}
    AxesOrigin -> {17, 12}
]
```



Dupa ce am construit toate interpolarile si aproximarile propuse urmeaza sa le afisam in acelasi grafic, iar apoi sa il centram in punctul reprezentand valoarea reala pe care o cautam ({17, 12}), adica numarul de faulturi comise si numarul de suturi spre poarta din primul meci de dupa grupe.



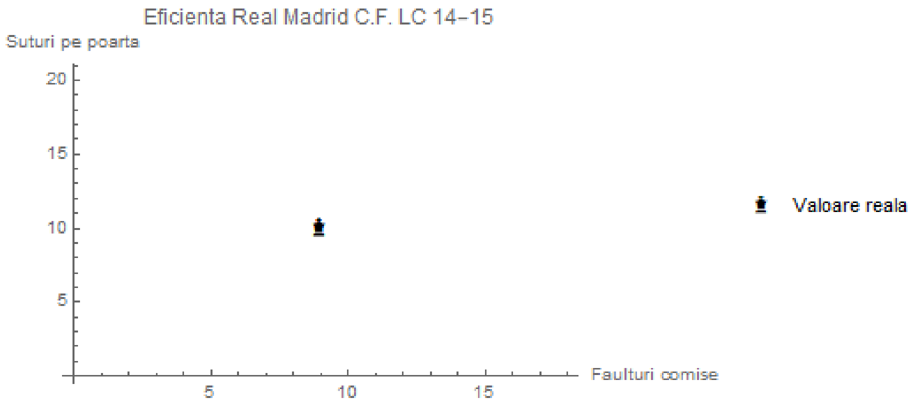
Valoarea aproximata este 9 si observam ca este o diferenta doar de 3 pana la valoarea reala 12, un rezultat bun.



2.3.5 Real Madrid C.F.

Ultima echipa ale carei rezultate dorim sa le aproximam in acest sezon competitional, 2014-2015, este Real Madrid C.F. Aceasta a jucat douasprezece meciuri in total, fiind eliminata in faza semifinalelor. Noi ne propunem sa aproximam numarul de suturi spre poarta din primul meci de dupa iesirea din grupe: FC Schalke 04 - Real Madrid C.F. 0 - 2. Real a comis 9 faulturi si a avut 10 suturi spre poarta in acest meci.

Valoarea reala pe care dorim sa o aproximam se gaseste in graficul de mai jos.



Suportul interpolarii

Din suportul interpolarii vor face parte unsprezece puncte ordonate dupa numarul de faulturi comise, adica primul termen al perechii.

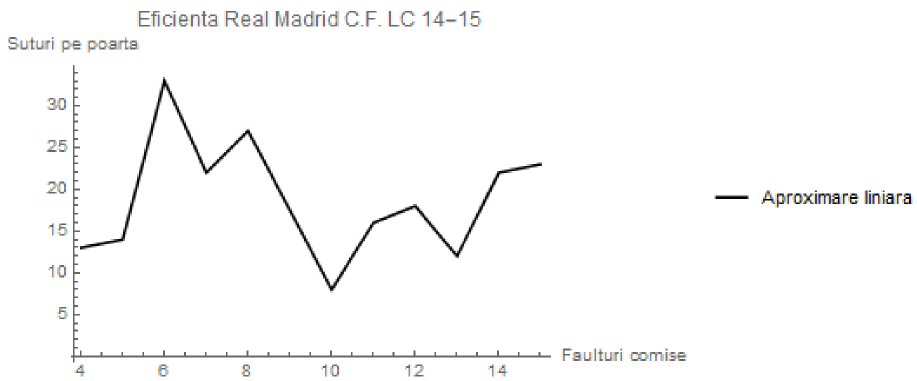
```
suportReal = {{4, 13}, {5, 14}, {6, 33}, {7, 22}, {8, 27}, {10,
8}, {11, 16}, {12, 18}, {13, 12}, {14, 22}, {15, 23}}
```

Interpolare liniara

Dorim sa realizam interpolarea liniara a punctelor din suport, iar acest lucru se poate obtine in Wolfram Mathematica astfel.

```
plotLiniarReal = ListPlot[
     suportReal,
     AxesOrigin->{3, 0},
     AxesLabel->{"Faulturi comise", "Suturi pe poarta"},
     PlotLabel->"Eficienta Real Madrid C.F. LC 14-15",
     Joined->True,
     PlotStyle->Black,
     PlotLegends->{"Aproximare liniara"}
]
```

Urmatoarea figura reprezinta graficul interpolarii liniare a punctelor ce apartin suportului interpolarii.

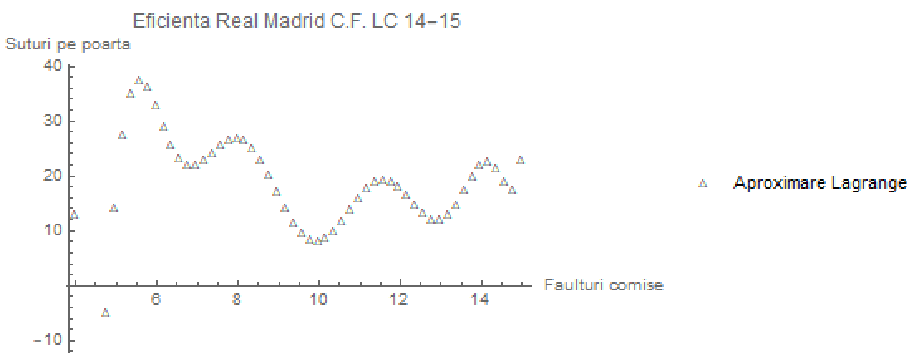


Interpolare Lagrange

Acum ne propunem sa construim polinomul Lagrange de interpolare care va avea grad maxim 10, deoarece este format folosind 11 puncte. Acesta va fi construit intre punctele 4 si 15, adica cea mai mica si cea mai mare valoare a numarului de faulturi pe care le-a obtinut Real Madrid C.F. intr-un meci din acest sezon competitional.

```
functieLagrangeReal = InterpolatingPolynomial[
     suportReal,
     x
]

puncteLagrangeReal = Table[
     functieLagrangeReal,
     {x, 4, 15, 0.2}
]

plotLagrangeReal = ListPlot[
     puncteLagrangeReal,
     DataRange -> {4, 15},
     AxesOrigin -> 3, 0,
     AxesLabel->{"Faulturi comise", "Suturi pe poarta"},
     PlotLabel->"Eficienta Real Madrid C.F. LC 14-15",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare Lagrange"},
     PlotMarkers -> {"\[EmptyUpTriangle]"}
]
```

Graficul reprezentativ pentru interpolarea Lagrange este urmatorul. Se poate observa ca pentru aceasta echipa oscilatiile polinomului sunt mult mai mici decat in alte exemple anterioare.



Interpolari de tip spline

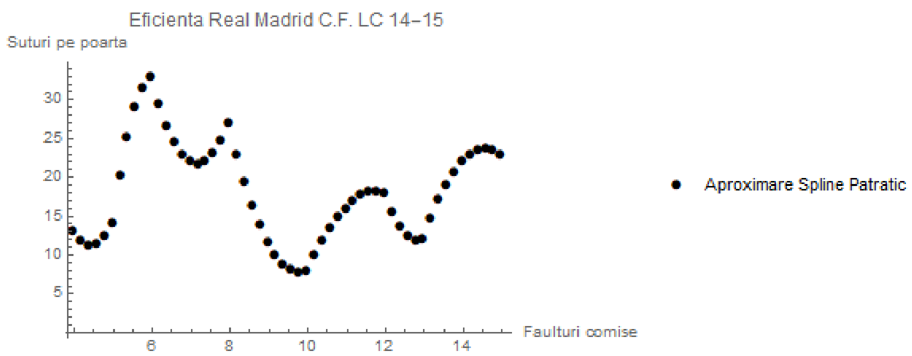
Construim cele doua interpolari spline, cu functii de imbinare patratice si cubice.

Incepem cu generarea punctelor pentru abordarea cu functii patratice, folosindu-ne de functia ”Interpolation” din Wolfram Mathematica.

```
functieSplinePatraticReal = Interpolation[
     suportReal,
     InterpolationOrder->2][x]
]

puncteSplinePatraticReal = Table[
     functieSplinePatraticReal,
     {x, 4, 15, 0.2}
]

plotSplinePatraticReal = ListPlot[
     puncteSplinePatraticReal,
     DataRange -> {4, 15},
     AxesOrigin -> {3, 0},
     AxesLabel->{"Faulturi comise", "Suturi pe poarta"},
     PlotLabel->"Eficienta Real Madrid C.F. LC 14-15",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare Spline Patratic"},
     PlotMarkers -> {"\[FilledCircle]"}
]
```

Graficul interpolarii spline cu functii de imbinare patratice se poate vedea in figura de mai jos.

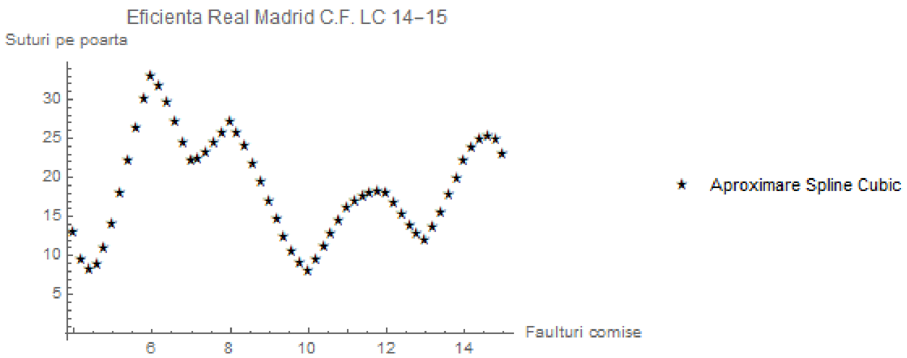
Observam ca foarte multe functii spline consecutive isi schimba convexitatea.



Construim si interpolarea spline cu functii de imbinare cubice si asteptam rezultate asemanatoare cu abordarea anterioara.

```
functieSplineCubicReal = Interpolation[
     suportReal,
     InterpolationOrder->3][x]
]

puncteSplineCubicReal = Table[
     functieSplineCubicReal,
     {x, 4, 15, 0.2}
]

plotSplineCubicReal = ListPlot[
     puncteSplineCubicReal,
     DataRange -> {4, 15},
     AxesOrigin -> {3, 0},
     AxesLabel->{"Faulturi comise", "Suturi pe poarta"},
     PlotLabel->"Eficienta Real Madrid C.F. LC 14-15",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare Spline Cubic"},
     PlotMarkers -> {"\[FivePointedStar]"}
]
```

Graficul interpolarii spline cu functii de imbinare cubice se poate vedea in figura de mai jos.



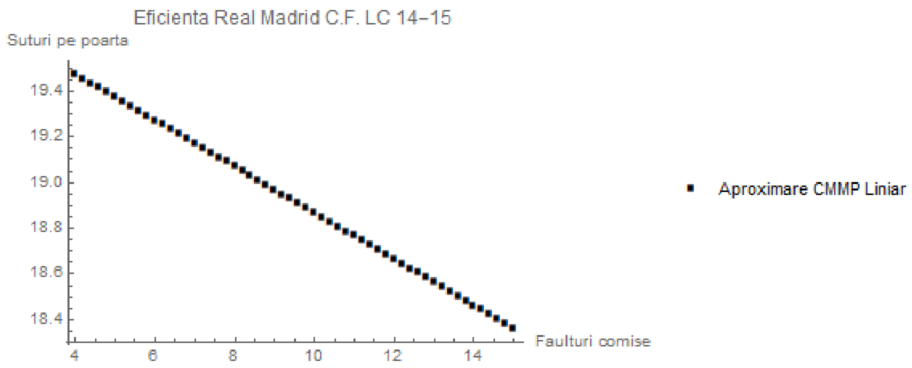
Aproximari in sensul CMMP

Pentru a aproxima cu un polinom liniar in sensul celor mai mici patrate e nevoie sa gasim un polinom pentru care suma distantelor de la punctele din suportul interpolarii la el este minima.

```
functieCMMPLiniarReal = Fit[suportReal, {1, x}, x]

puncteCMMPLiniarReal = Table[
     functieCMMPLiniarReal,
     {x, 4, 15, 0.2}
]

plotCMMPLiniarReal = ListPlot[
     puncteCMMPLiniarReal,
     DataRange -> {4, 15},
     AxesOrigin -> {3, 0},
     AxesLabel->{"Faulturi comise", "Suturi pe poarta"},
     PlotLabel->"Eficienta Real Madrid C.F. LC 14-15",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare CMMP Liniar"},
     PlotMarkers -> {"\[FilledSquare]"}
```

```
]
```

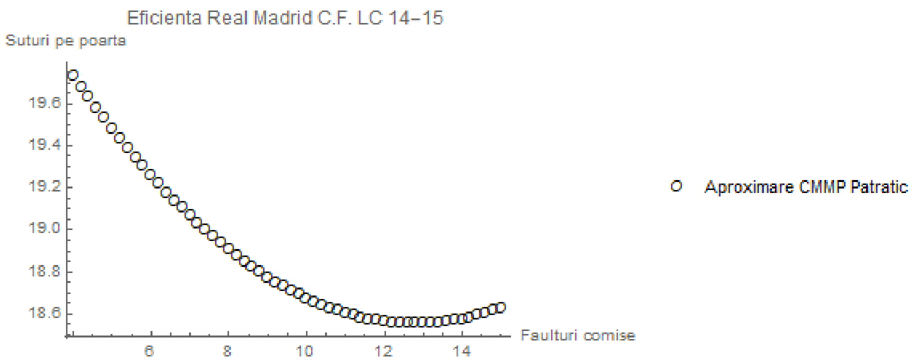
Graficul aproximarii liniare este ca cel de mai jos.



Aproximarea printr-un polinom patratic ales in sensul celor mai mici patrate care sa treaca printre toate punctele din suportul interpolarii se realizeaza in Wolfram Mathematica astfel.

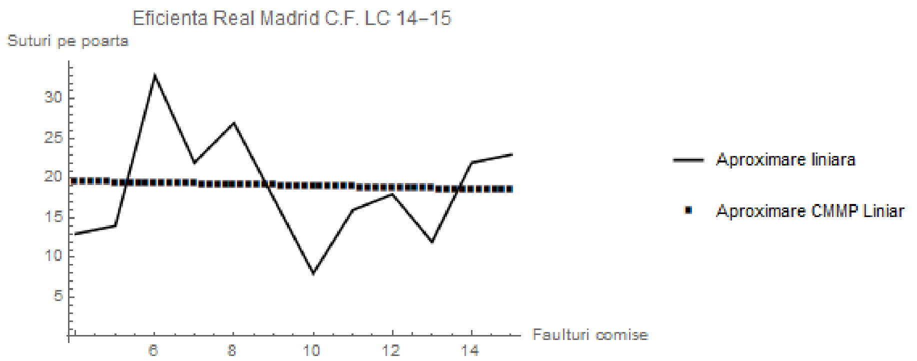
```
functieCMMPPatraticReal = Fit[suportReal, {1, x, x^2}, x]
puncteCMMPPatraticReal = Table[
     functieCMMPPatraticReal,
     {x, 4, 15, 0.2}
]

plotCMMPPatraticReal = ListPlot[
     puncteCMMPPatraticReal,
     DataRange -> {4, 15},
     AxesOrigin -> {3, 0},
     AxesLabel->{"Faulturi comise", "Suturi pe poarta"},
     PlotLabel->"Eficienta Real Madrid C.F. LC 14-15",
     PlotStyle->Black,
     PlotLegends -> {"Aproximare CMMP Patratic"},
     PlotMarkers -> {"O"}
]
```

Graficul aproximarii patratice este cel de mai jos.

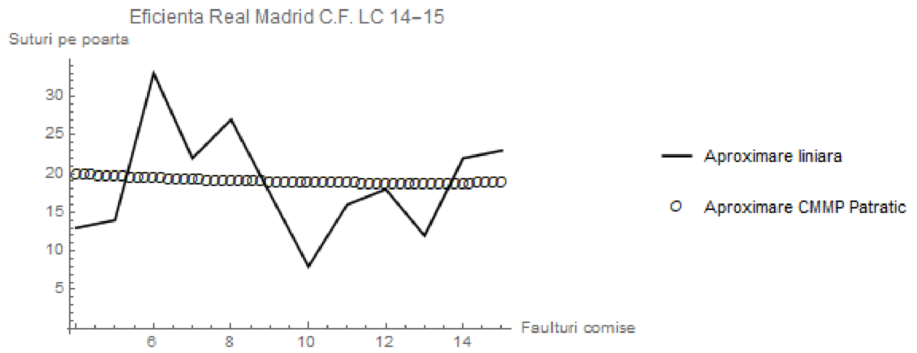


Urmeaza sa suprapunem graficul interpolarii liniare cu graficele aproximarilor in sensul celor mai mici patrate.

```
Show[plotLiniarReal, plotCMMPLiniarReal]
```

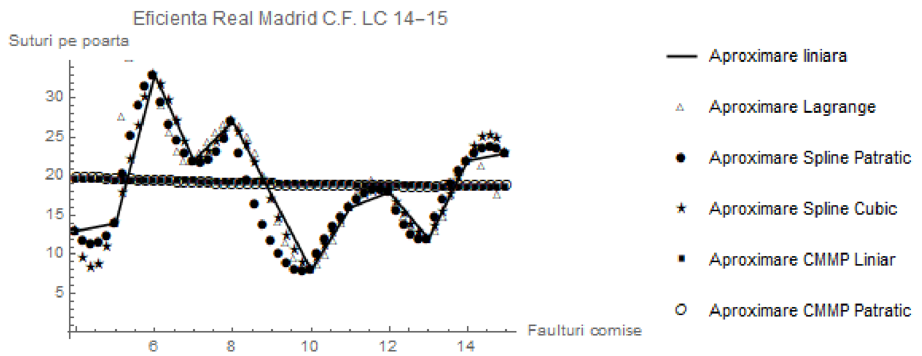


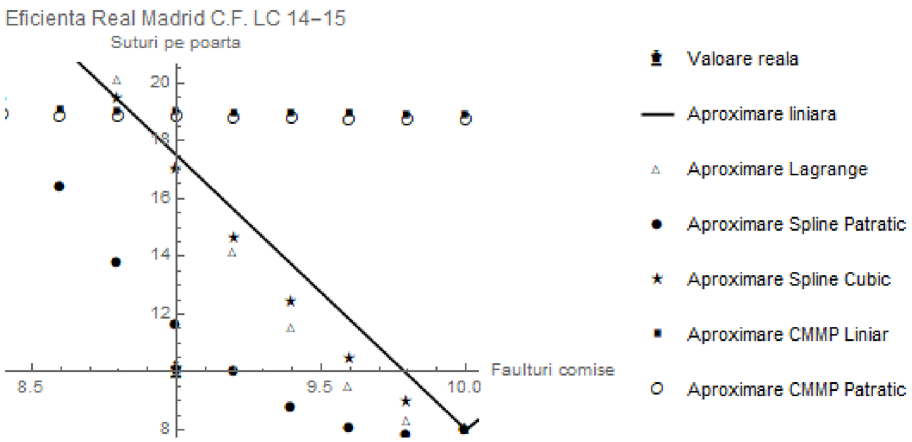

```
Show[plotLiniarReal, plotCMMPLiniarReal]
```



Dupa ce am construit toate interpolarile si aproximarile propuse urmeaza sa le afisam in acelasi grafic, iar apoi sa il centram in punctul reprezentand valoarea reala pe care o cautam ({9, 10}), adica numarul de faulturi comise si numarul de suturi la poarta din meciul dorit.

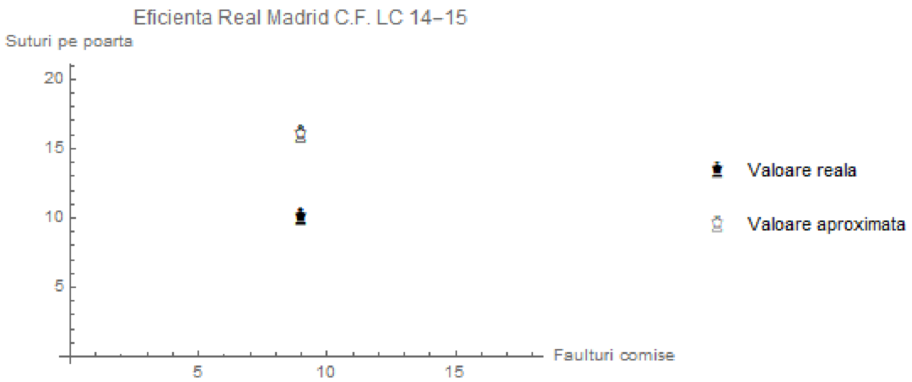
```
Show[
     plotLiniarReal,
     plotLagrReal,
     plotSplinePatraticReal,
     plotSplineCubicReal,
     plotCMMPLiniarReal,
     plotCMMPPatraticReal,
     PlotRange -> {{8, 10}, {8, 20}}
     AxesOrigin -> {9, 10}
]
```





Vom folosi toate rezulatele obtinute, astfel, in punctul de coordonata $x = 9$, adica valoarea reala a posesiei din meciul urmarit, vom obtine $y = 16$. Aceasta valoare este destul de indepartata de valoarea reala si urmarind graficul anterior observam ca acest meci a fost atipic pentru Real Madrid C.F. in sezonul 2014-2015.

Valoarea reala si valoarea aproximata puse una langa alta arata ca mai jos.



Bibliografie

[1] Numerical Methods, 4th - J. Douglas Faires, Richard L. Burden, ISBN 13: 9780495114765, Publisher: Cengage Learning, 2014