Факултет инжењерских наука

Универзитета у Крагујевцу

Назив студијског програма: Машинско инжењерство

Ниво студија: Основне академске студије

Модул: Примењена механика и аутоматско управљање

Предмет: Аутоматско управљање

Број индекса: 507/2012

Огњен Г. Думбеловић

Аутоматски систем за процену времена трајања дијализе

Завршни рад

Комисија за преглед и одбрану:

1. **Проф. др Александар Пеулић – ментор**
2. **Проф. др Весна Ранковић**
3. **Проф. др Петар Тодоровић**

Датум одбране:_____

Оцена:_____

Изјава о самосталној изради рада

Изјављујем да сам овај завршни рад урадио самостално, служећи се стеченим знањем и искуством, као и наведеном литературом.

Број индекса	Име и презиме
507/2012	Огњен Думбеловић

Резиме

Циљ овог рада је реализација аутоматизованог система за процену параметара током процеса дијализе и предикцију времена трајања процеса. За реализацију су коришћене реалне вредности мерења и имплементирани су нелинеарни интерполациони полиноми Лагранжов и Њутнов на микрорачунару *Raspberry Pi*. Моделирајући процесе коришћењем програмског језика *Python,* написани су алгоритми за поменуте методе интерполације и извршена је њихова реализација, упоређивање као и процена квалитета самих метода.

Кључне речи: апроксимација, интерполација, Лагранжов интерполациони полином, Њутнов интерполациони полином, *Raspberry Pi, Python*

Abstract

The purpose of this paper is the realization of an automatic control system, used for the determination of certain parameters within the process of dialysis and predicting the duration of said process. Real values were used during this realization, implemented with Newton and Lagrange non-linear interpolation polynomials on a Raspberry Pi microcomputer. Using the Python programming language, with which the process is being modeled, algorithms are formed in respect to the methods mentioned above as is the realization, comparison and quality assessment of said methods.

Keywords: approximation, interpolation, Lagrange interpolating polynomial, Newton interpolating polynomial, Raspberry Pi, Python

Садржај

1. Идентификација и апроксимација 6
1.1. Проблеми идентификације 6
1.2. Проблеми апроксимације 7
1.3. Линеарне апроксимационе функције 9
1.4. Нелинеарне апроксимационе функције 10
1.5. Критеријуми апроксимације 10
2. Интерполација 11
2.1. Лагранжов интерполациони полином 12
2.2. Њутнов интеполациони полином 13
2.4. Колико је добар интерполациони полином? 15
3. Практична реализација 16
3.1. Дефинисање проблема 16
3.2. Мерење потребних параметара 16
3.3. Писање алгоритма у програмском језику "*Python*" 17
3.3.1. Алгоритам Лагранжове интерполације 18
3.3.2. Алгоритам Њутнове интерполације 20
3.4. Приказ и провера добијених резултата 23
4. Закључак 26
5. Литература 28

1. Идентификација и апроксимација

Велики број проблема који се јављају током процеса пројектовања аутоматских система предикције односи се на методе идентификације и апроксимације [1].

Појам идентификације се широко користи у теорији система управљања и подразумева процес креирања математичког модела динамичког система на основу прикупљених података, обично на основу мерења улазних и излазних сигнала система. Примењује се када посматрани систем није довољно проучен или описан и када за њега не постоји позната и одређена математичка функција која повезује излазе са улазом. Често се под појмом идентификације подразумева само процес одређивања параметара изабраног математичког модела за посматрани диманички систем.

Типичан задатак у теорији апроксимације је одређивање функција дате класе тако да добијена апроксимативна крива на посматраном интервалу одступа што је мање могуће од одређене задате криве. Друга велика класа проблема у теорији апроксимације је одређивање функције апроксимативне криве на основу познатих вредности, променљивих и функције у коначном броју дискретних тачака.

1.1. Проблеми идентификације

Идентификација се бави проблемима дефинисања погодних математичких модела динамичких система, по правилу објекта управљања, на основу експерименталних података и анализе истих. Променљиве које описују систем најпре се поделе на улазне и излазне променљиве. Мерењем вредности улазних и излазних променљивих добијају се подаци на основу којих треба идентификовати систем, тј. развити одговарајући математички модел који описује динамику система и одредити његове параметре.

У општем случају процес идентификације обухвата решавање следећих основних задатака:

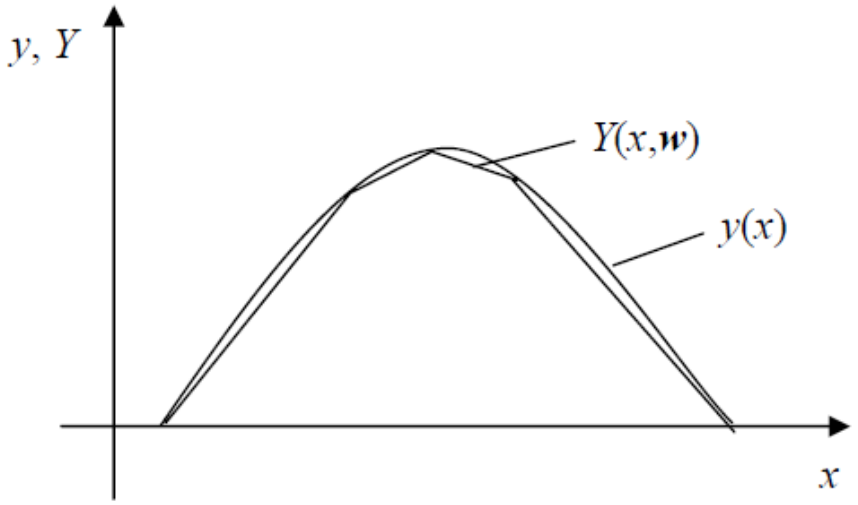
- избор класе математичких модела
- избор класе улазних сигнала
- избор критеријума за оцену колико модел одговара систему
- развој и примена алгоритма за решавање

У ужем смислу, под идентификацијом се подразумева решавање задатка одређивања оптималних параметара изабраног математичког модела, где се претпоставља да су претходно усвојени класа модела и критеријум за оцену квалитета апроксимације система моделом.

1.2. Проблеми апроксимације

У теорији апроксимације развијају се методе апроксимације или интерполације непрекидне скаларне функције $y(x)$ помоћу апроксимативне функције $Y(w,x)$ која има одређени број параметара w који припада неком скупу. За изабрану функцију проблем је наћи вектор параметара w који даје најбољу могућу апроксимацију функције $y(x)$ на посматраном скупу.

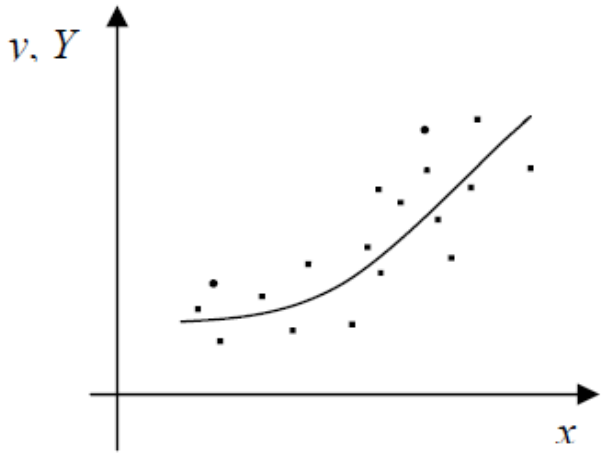
Постоје два тима апроксимације. Први се односи на случајеве када задату криву $y(x)$ треба апроксимирати функцијама дате класе или њиховим комбинацијама, тако да добијена апроксимативна крива $Y(w,x)$ на посматраном интервалу одступа од задате што је могуће мање. У класичној теорији апроксимације као апроксимативне функције уобичајено се користе комбинације линеарних функција, синусних функција итд.



Слика 1. Апроксимација нелинеарне криве линеарним одсечцима

На слици 1 илустрована је апроксимација једне нелинеарне криве помоћу неколико линеарних функција. Лако се уочава да квалитет апроксимације дате нелинеарне функције зависи од броја апроксимирајућих линеарних функција.

Други тип решавања проблема односи се на случајеве када су познате реализације посматраних променљивих у одређеним дискретним тачкама $(x_i, y(x_i))$ слика 2. где се вредности обично добијају мерењем реалних величина посматраног система. У том случају треба одредити апроксимативну/интерполациону криву $Y(w,x)$ и њене параметре тако да збир растојања апроксимираних вредности $Y(w,x)$ од реализација $y(x)$ у посматраним тачкама буде минималан.

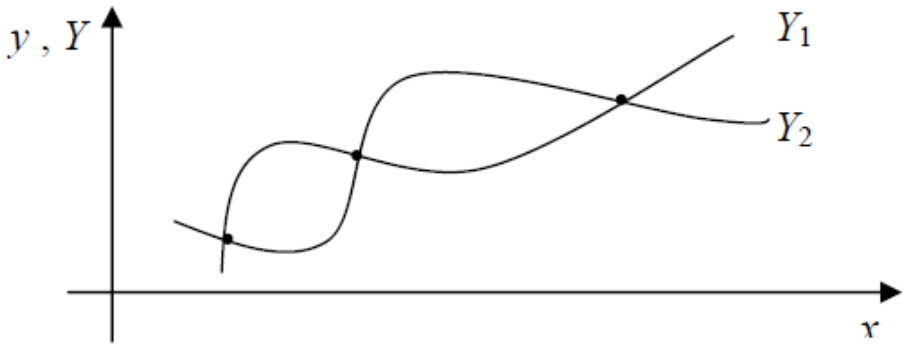


Слика 2. Апроксимација на основу узорака

Од поменута два типа решавања проблема апроксимације, други тип решавања се у пракси више користи. Најчешће се јавља код мерења различитих физичких величина, јер, осим измерених података, покушавамо апроксимирати и податке који се налазе између измерених тачака. Грешке се могу јавити и код начина мерења података па тако постоје и одређени модели/технике за смањење тако насталих грешака.

Функција се бира према природи модела, али на тај начин да буде релативно једноставна за рачунање. Када функцију добијемо у канонском облику онда можемо да кажемо да смо изабрали општи облик апроксимативне функције. Облике апроксимативних функција грубо можемо поделити на линеарне и нелинеарне.

Треба имати на уму да је задатак апроксимације функције на основу одређених узорака, тј. на основу скупа дискретних вредности, математички слабо постављен проблем јер не постоји довољно података да се реконструише пресликавање у областима за које не постоје подаци. Другим речима, можемо конструисати више међусобно различитих апроксимација које су када посматрамо укупну грешку, подједнако добре за податке којима се располаже. Те разлике се нарочито испољавају у областима за које се не познају оригинални подаци, слика 3.



Слика 3. Две апроксимације једнаке укупне грешке

Поред тога подаци по правилу нису апсолутно тачни, већ садрже неку врсту статистичке грешке. То ствара додатне проблеме у апроксимацији и потребу да се на почетку поставе одређени услови или претпоставке како би задатак апроксимације био добро постављен.

На задацима идентификације и апроксимације лепо се уочавају важност и улога избора класе математичког модела и апроксимативних функција као и избор критеријума којим се оцењује квалитет и ваљаност решења.

Нема потребе посебно објашњавати колико је важно изабрати апроксимативну функцију Y која може представити y што је могуће боље. Било би скоро узалудно тражити оптималне вредности параметара криве $Y(w,x)$ у случају када она може да оствари само сиромашну представу функције $y(x)$, нпр. ако јако нелинеарну функцију покушавамо да апроксимирамо линеарном функцијом.

Из тог разлога у решавању задатака апроксимације треба разликовати следеће главне проблеме [2]:

- Проблем који апроксимацију користи тј. које класе функција $y(x)$ могу да се ефективно апроксимирају којим апроксимативним функцијама $Y(w,x)$. Ово је проблем представљања или репрезентације

- Проблем који алгоритам користи за налажење оптималних вредности параметара w за дати избор Y, укључујући у овај проблем и избор критеријума за оцену квалитета решења

1.3. Линеарне апроксимационе функције

Општи облик линеарне апроксимационе функције је:

$$\varphi(x) = a_0\varphi_0(x) + a_1\varphi_1(x) + \ldots + \varphi_m(x),$$

где су $\varphi_0, \ldots, \varphi_m$ већ познате функције. Одавде можемо приметити да линеарност независи од облика функције φ, него од параметара a_k које треба одредити. Предност овог облика апроксимационе функције је да се одређивање параметара a_k углавном своди на решавање система линеарних једначина.

Најчешћи облици линеарних апроксимационих функција су:

1. Алгебарски полиноми

$$\varphi_k(x) = x^k, k = 0, \ldots, m$$
$$\varphi_k(x) = a_0 + a_1x + \cdots + a_mx^m$$

2. Тригонометријски полиноми, погодни за апроксимацију периодичних функција, често се користе код моделирања сигнала

$$\{1, cosx, sinx, cos2x, sin2x, \ldots\}$$

3. "Сплајн" функције, углавном се користе због добрих карактеристика када се у обзир узимају грешка и крива апроксимационе функције

$$\varphi|_{[x_k x_k - 1]} = p_k, k = 1,2, \ldots n$$

где су p_k полиноми најчешће степена 1,2,3 или 5.

1.4. Нелинеарне апроксимационе функције

Најчешћи облици нелинеарних апроксимационих функција су:

1. Експоненцијалне апроксимације

$$\varphi(x) = c_0 e^{b_0 x} + c_1 e^{b_1 x} + \cdots + c_r e^{b_r x},$$

које имају $n = 2r + 2$ независна параметра и имају примену код процеса праћења наталитета и морталита, биологији, економији, медицини итд.

2. Рационалне апроксимације

$$\varphi(x) = \frac{b_0 + b_1 x + \cdots + b_r x^r}{c_0 + c_1 x + \cdots + c_s x^s)}$$

које имају $n = r + s + 1$ независна параметра. Због могућности проширивања разломка увек се "фиксира" један од параметара b_i или c_i што се обично закључује на основу природе посматраног модела.

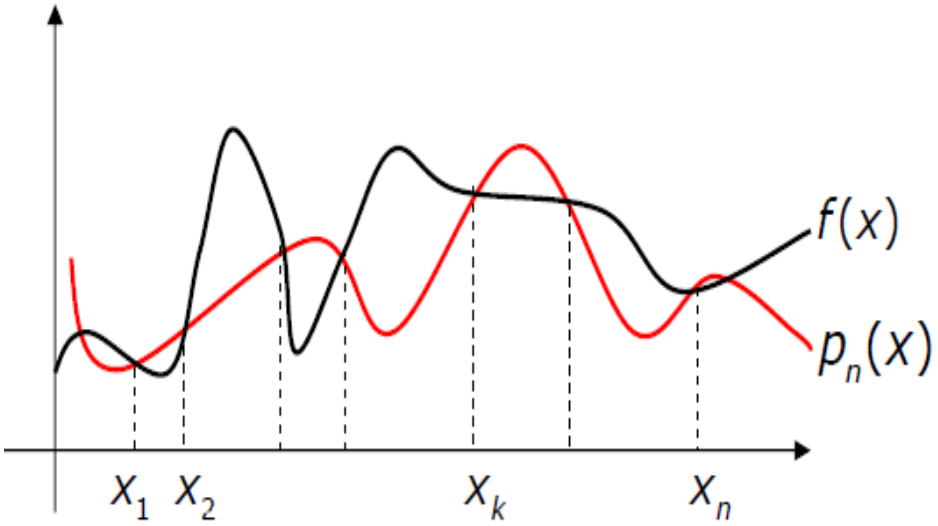
1.5. Критеријуми апроксимације

Апрокцимационе функције бирају се тако да “најбоље” задовоље постављене услове. Најчешћи захтеви су да граф апроксимационе функције пролази одређеним тачкама тј. да интерполира функцију у тим тачкама или да је одступање полазне од апроксимационе функције минимално, односно да се грешка минимализује.

Ово су општи захтеви апроксимације, међутим даљи избор и конструкција апроксимационе криве зависе од различитих параметара посматраног проблема. Од многобројних приступа интерполацији у даљем делу рада биће описани они који се најчешће користе и који највише одговарају практичној реализацији проблема који разматрамо у овом раду.

2. Интерполација

Интерполација [3] је захтев да се вредности дате функције и интерполационог полинома подударају на неком коначном скупу тачака или аргумената. Те тачке обично називамо чворовима интерполације. Овом захтеву се може додати захтев да се у чворовима, осим функцијских вредности поклапају и вредности неких деривација. Због јединствености полиномске функције која задовољава полазни услов, најчешћа намера је да се формира функција полиномског типа која интерполира полазну функцију. Полином и интерполациона функција се, као што је поменуто, поклапају у чворовима интерполације док у осталим чворовима то не мора да буде случај (слика 4).



Слика 4. Графици интерполационог полинома и дате функције

Ход интерполације представља растојање између суседних чворова $h = d(x_i, x_{i+1})$, тако да према ходу интерполационе мреже делимо на:

- еквидистантне
- променљиве

У пракси је редак случај да се тражена вредност поклапа са интерполационим чворовима, тако да ћемо у раду обратити пажњу на полиноме који су карактеристични за променљиве интерполационе мреже. Такође, у раду ћемо кроз практичну реализацију интерполационих метода приказати и упоредити њихову верност и квалитет апроксимације.

2.1. Лагранжов интерполациони полином

У нумеричкој анализи, Лагранжов полином се користи у области полиномске интерполације. Методу је објавио Јозеф Луис Лагранж 1795. године по коме је метода и названа.

Како би нашли коефицијенте интерполационог полинома, не морамо решавати систем линеарних једначина. Интерполациони полином p_n можемо написати коришћењем тзв. Лагранжове базе $\{l_k, k = 0, \dots, n\}$:

$$p_n(x) = \sum_{k=0}^{n} f_k \, l_k(x)$$

при чему је:

$$l_k(x) = \frac{(x - x_0) \dots (x - x_{k-1})(x - x_{k+1}) \dots (x - x_n)}{(x_k - x_0) \dots (x_k - x_{k-1})(x_k - x_{k+1}) \dots (x_k - x_n)} = \prod_{\substack{i=0 \\ i \neq k}}^{n} \frac{x - x_i}{x_k - x_i} = \frac{w_{k(x)}}{w_{k(x_k)}}$$

за $k = 0, \dots, n$. Овај облик полинома представља Лагранжов интерполациони полином, где су полиноми l_k степена n па је полином p_n највишег степена n. Поред тога важи:

$$l_{k(x_i)} = \begin{cases} 0, & \text{за } i \neq k \\ 1, & \text{за } i = k \end{cases}$$

Сведенији облик Лагранжовог полинома можемо добити уколико ову једнакост заменимо у полиному.

$$p_{n(x_k)} = f_k$$

Полином такође можемо записати у облику:

$$w(x) = \prod_{k=0}^{n} (x - x_k)$$

па је

$$l_k = \frac{w(x)}{(x - x_k) w_{k(x_k)}}$$

$$p_{n(x)} = w(x) \sum_{k=0}^{n} \frac{f_k}{(x - x_k) w_k(x_k)}$$

Овај облик се може користити за рачунање вредности полинома у тачкама различитим

од x_k за $k = 0, \dots, n$. Мана овог полинома, уколико посматрамо са аспекта броја операција или оптимизације кроз алгоритам, јесте у томе што је број операција n^2. Ипак, у пракси се ређе користи овај облик полинома у поређењу са Њутновим обликом интерполационог полинома који ће бити објашњен у даљем делу рада.

2.2. Њутнов интеполациони полином

Другачији начин приступа интерполацији дао је Исак Њутн својим интерполационим полиномом са подељеним разликама [4]. Ова метода представља основу за већину других, комплекснијих метода које се баве проблемима нумеричке анализе.

Као што је већ поменуто, Лагранжов интерполациони полином не представља најбољу методу интерполације због проблема тачности (прецизности) добијених апроксимационих вредности при већим степенима полинома. Овај проблем се јавља из разлога што сваки интерполациони полином морамо рачунати из почетка.

Њутнов облик представља полином где је, за разлику од Лагранжовог, много лакше повећати степен односно додати чворове интерполације не рачунајући полином из почетка већ поступком рекурзије.

Нека је p_n полином који интерполира функцију f у тачкама $x_k, k = 0, \dots, n-1$. Нека p_n интерполира функцију f још и у тачки x_n. Полином p_n тада можемо написати у облику:

$$p_n(x) = p_{n-1}(x) + c(x)$$

где је c корекција, а полином степена n. Такође мора важити:

$$c(x) = p_{n(x_k)} - p_n - 1(x_k) = f(x_k) - f(x_k) = 0, \qquad k = 0, \dots, n-1$$

Примећујемо да су x_k нуле функције c па корекцију можемо написати у облику:

$$c(x) = a_n(x - x_0) \dots (x - x_{n-1})$$

Из услова интерполације да је $p_n(x_n) = f(x_n)$ добијамо:

$$f(x_n) = p_n(x_n) = p_n - 1(x_n) + c(x_n) = p_{n-1}(x_n) + a_n(x_n - x_0) \dots (x_n - x_{n-1})$$

одакле лако можемо израчунати главни(водећи) коефицијент a_n полинома c

$$a_n = \frac{f(x_n) - p_{n-1}(x_n)}{(x_n - x_0) \dots (x_n - x_{n-1})} = \frac{f(x_n) - p_{n-1}(x_n)}{w(x_n)}$$

У овом тренутку имамо све потребне елементе за рачунање $p_n(x)$ у било којој тачки x. Коефицијент a_n представља функцију чворова $x_0, \dots, x_n$ па га можемо записати:

$$a_n = f[x_0, x_1, \dots, x_n]$$

Одмах након овог корака следи рекурзивна формула за добијање интерполационог

полинома који је за један степен виши од претходног:

$$p_n(x) = p_{n-1}(x) + (x - x_0) \dots (x - x_{n-1}) f[x_0, \dots, x_n]$$

Поред ове рекурзивне формуле потребно је још утврдити почетак и формулу за рекурзивно рачунање подељених разлика. Кратким поступком можемо да изведемо ову формулу:

$$\begin{aligned} f[x_1, \dots, x_n] &= \sum_{k=1}^{n} \frac{f(x_k)}{(x_k - x_1) \dots (x_k - x_{k-1})(x_k - x_{k+1}) \dots (x_k - x_n)} \\ &= \sum_{k=1}^{n-1} \frac{f(x_k)(x_k - x_0)}{(x_k - x_0) \dots (x_k - x_{k-1})(x_k - x_{k+1}) \dots (x_k - x_n)} \\ &+ \frac{f(x_n)(x_n - x_0)}{(x_n - x_0) \dots (x_n - x_{n-1})} \end{aligned}$$

$$\begin{aligned} f[x_0, \dots, x_{n-1}] &= \sum_{k=0}^{n-1} \frac{f(x_k)}{(x_k - x_0) \dots (x_k - x_{k-1})(x_k - x_{k+1}) \dots (x_k - x_{n-1})} \\ &= \sum_{k=1}^{n-1} \frac{f(x_k)(x_k - x_n)}{(x_k - x_0) \dots (x_k - x_{k-1})(x_k - x_{k+1}) \dots (x_k - x_n)} \\ &- \frac{f(x_0)(x_n - x_0)}{(x_0 - x_1) \dots (x_0 - x_n)} \end{aligned}$$

Одузимањем добијамо тражену формулу:

$$\begin{aligned} f[x_1, \dots, x_n] - & f[x_0, \dots, x_{n-1}] \\ &= \sum_{k=1}^{n-1} \frac{f(x_k)(x_n - x_0)}{(x_k - x_0) \dots (x_k - x_{k-1})(x_k - x_{k+1}) \dots (x_k - x_n)} \\ &+ \frac{f(x_n)(x_n - x_0)}{(x_n - x_0) \dots (x_n - x_{n-1})} + \frac{f(x_0)(x_n - x_0)}{(x_0 - x_1) \dots (x_0 - x_n)} \\ &= (x_n - x_0) \sum_{k=0}^{n} \frac{f(x_k)}{w_k(x_k)} = (x_n - x_0) f[x_0, \dots, x_n] \end{aligned}$$

Преостао је још део за одређивање почетка рекурзије за подељене разлике. Знамо да је константа која пролази кроз тачку $(x_0, f(x_0))$ интерполациони полином степена 0 тада важи $a_0 = f[x_0] = f(x_0)$, односно:

$$f[x_k] = f(x_k)$$

Сада можемо да формирамо таблицу подељених разлика:

x_k	$f[x_k]$	$f[x_k, x_{k+1}]$	$f[x_k, x_{k+1}, x_{k+2}]$	$\cdots$	$f[x_0, \ldots, x_n]$
x_0	$f[x_0]$				
		$f[x_0, x_1]$			
x_1	$f[x_1]$		$f[x_0, x_1, x_2]$		
		$f[x_1, x_2]$		$\ddots$	
$\vdots$	$\vdots$	$\vdots$	$\vdots$		$f[x_0, \ldots, x_n]$
		$f[x_{n-2}, x_{n-1}]$		$\iddots$	
x_{n-1}	$f[x_{n-1}]$		$f[x_{n-2}, x_{n-1}, x_n]$		
		$f[x_{n-1}, x_n]$			
x_n	$f[x_n]$				

Коначно уколико усвојимо рекурзију и облик полинома c добијамо коначни облик Њутновог интерполационог полинома:

$$p_n(x) = f[x_0] + (x - x_0)f[x_0, x_1] + (x - x_0)(x - x_1)f[x_0, x_1, x_2] + \cdots + (x - x_0) \ldots (x - x_{n-1})f[x_0, x_1, \ldots, x_n]$$

2.4. Колико је добар интерполациони полином?

У пракси се обично користе интерполациони полиноми нижих степена, најчешће до 5. Они се примењују из разлога што код неких функција за одређени избор чворова интерполације повећавање степена интерполационог полинома може довести до знатног повећања грешака. Због тога се уместо виших степена интерполационог полинома, у пракси користи интерполација по деловима коришћењем полинома нижег степена.

3. Практична реализација

У оквиру завршног рада применом рачунара "*Raspberry Pi*" [5] [6]вршена је симулација дигиталног управљања на конкретном проблему код процеса дијализе. Решавање овог проблема можемо поделити у неколико корака:

- Дефинисање проблема
- Мерење потребних параметара
- Одређивање математичког модела система
- Писање алгоритма у програмском језику "*Python*"
- Приказ и провера добијених резултата

3.1. Дефинисање проблема

Систем који моделирамо, пре свега морамо добро да опишемо. Код процеса дијализе пацијената, на уређајима за дијализу подеси се колико је течности захтевано да уређај одвади током процеса. Тада се уопште не зна колико ће сам процес да траје, а како се мерења потребних параметара не могу извршити у стационарном стању, није згодно да се пацијент у кратким временским интервалима помера да би се извршило мерење тежине. Како би смо смањили интервале мерења и направили предикцију, одређена подешавања параметара могу да се задају при самом почетку дијализе и систем би онда предвидео време дијализе као и процену параметара. Тако би систем сам успоставио подешавање вађења течности у зависности од тренутне адаптације пацијента што управо представља циљ овог рада.

Параметри који се посматрају су време процеса, ТТ(телесна тежина) и БИ(*body index*) који је преко висине повезан са телесном тежином. Крајња ТТ се пројектује, задаје и прати у ком тренутку систем достиже задату вредност. Циљ је да се на основу што мање мерења података ТТ и БИ направи што боља предикција за које време ће систем да достигне задату телесну тежину.

3.2. Мерење потребних параметара

За процену тачности модела, односно његових предикција, у раду су за потребне параметре узете табеле већ прикупљених вредности мерења код пацијената у процесу дијализе [7][8][9] у различитим временским интервалима, сврстаних по датумима, Табела 1.

	5.avg		7.avg		12.avg		14.avg		19.avg		21.avg	
vreme	TT	BI	TT	BI	TT	BI	TT	BI	TT	BI	TT	BI
uključenje	63,5	509	61,6	521	62,1	513	61,4	545	66	413	63	551
15min	63,5	513	61,6	534	62,5	512	61,4	554	65,7	521	62,8	591
30min	63,2	523	61,4	539	62,4	626	61,2	562	65,2	507	62,5	621
45min	62,9	533	61,3	542	62	535	61	570	65,1	510	62,4	625
60min	62,7	538	61	555	61,5	529	61,1	579	65	534	62,1	634
75min	62,4	546	61	565	61,6	521	61	584	64,8	549	61,9	611
90min	62,2	555	60,7	559	61,3	541	61,8	588	64,5	553	62	600
105min	62	560	60,4	563	61,2	556	61,3	596	64,45	560	61,6	657
120min	61,7	565	60,2	571	61	568	60,6	806	64,2	569	61,4	657
135min	61,5	569	60	577	60,7	585	60,2	735	63,9	568	61,2	664
150min	61,2	581	59,8	583	60,4	592	59,7	606	63,6	715	61	678
165min	61	594	59,6	592	60,3	590	59,6	781	63,3	635	60,9	661
180min	60,7	593	59,4	598	60,15	730	59,5	776	63,7	581	60,7	708
195min	60,4	600	59,1	605	60,05	712	59,4	769	61,9	595	60,5	703
210min	60,1	608	58,9	622	59,7	603	59,1	956	61,7	679	60,3	711
225min	59,9	618	58,7	610	59,2	620	58,9	846	61,5	607	60,1	716
240min	59,7	626	58,5	621	59	643	58,8	768	62,3	686	60	680

Табела 1. Вредности измерених параметара ТТ и БИ процеса дијализе

3.3. Писање алгоритма у програмском језику "*Python*"

Након посматрања параметара, прикупљања података и математичке анализе, решавамо проблем писањем алгоритма у програмском језику "*Python*" који је у *Raspberry Pi*-ју смештен у одељак "*Programming"* као на слици 5[10].

Слика 5. Директоријум програмског језика Python

Алгоритми у овом раду представљају имплементацију већ описаних модела Лагранжовог и Њутновог интерполационог полинома. Конкретно на нашем проблему, у оба алгоритма, рађене су две интерполације, за вредност времена и параметра ТИ као и за параметар БИ. Кроз излаз алгоритма се генерише табела међусобне зависности параметара током времена, чије вредности касније можемо упоређивати са измереним и

тиме оценити квалитет интерполационог модела као и самог алгоритма. Још један начин за евалуацију самог модела јесте кроз анализу и тестирање алгоритма кроз претходне процесе дијализе.

3.3.1. Алгоритам Лагранжове интерполације

```
# Unos pocetnih vrednosti merenja, njihovo dodavanje u niz i ispis na terminal
# Unos vrednosti krajnje TT
import math

x=[]
y=[]
z=[]

print("============================================================")
n=int(input("Unesi broj pocetnih merenja: "))
print("Unesi vrednosti merenja TT i BB iz tabele:")
for i in range(0,n):
    xtemp=float(input("vreme:"))
    x.append(xtemp)
    ytemp=float(input("TT:"))
    y.append(ytemp)
    ztemp=float(input("BI:"))
    z.append(ztemp)

print("============================================================")
print("Uneta tabela je:")
print("vreme   TT     BI")
for i in range(0,n):
    print(x[i],y[i],z[i])
print("============================================================")
tt=float(input("Unesi vrednost krajnje TT: "))

# Lagranzov interpolacioni polinom vreme-TT
s=1
float(s)
t=1
float(t)
k=0
float(k)

m=n
while(y[m-1]>=tt):
    x.append(x[m-1]+15)
    for i in range(0,n):
        s=1
        t=1
        for j in range(0,n):
            if (i-j!=0):
                s*=x[m]-x[j]
                t*=x[i]-x[j]
        k+=(s/t)*y[i]
    y.append(k)
    s=1
    t=1
    k=0
    m=m+1
```

```
# Lagranzov polinom za vreme-BI
s=1
t=1
k=0
for m in range(n,m):
    for i in range(0,n):
        s=1
        t=1
        for j in range(0,n):
            if (i-j!=0):
                s*=x[m]-x[j]
                t*=x[i]-x[j]
        k+=(s/t)*z[i]
    z.append(k)
    s=1
    t=1
    k=0

# Provera koja je vrednost bliza
if (abs(y[m]-tt)>abs(y[m-1]-tt)): w=m-1
else: w=m

#Ispis generisane tabele i rezultata programa na terminal i u datoteku
f=open("Datoteka.txt","w")

print("============================================================")
print("Generisana tabela:")
f.write("============================================================\n")
f.write("Generisana tabela:\n")

print("vreme  TT     BI")
f.write("vreme  TT     BI\n")
for i in range(0,m+1):
    print(x[i],y[i],z[i])
    f.write("%s   %s   %s\n" %(x[i],y[i],z[i]))
    if (i==m-1):
        print("------------------------------------------------------------")
        f.write("------------------------------------------------------------\n")
f.write("============================================================\n")
print("============================================================")
print("Sistem ce dostici zadatu tezinu za: %f minuta" %x[w])
f.write("Sistem ce dostici zadatu tezinu za: %s minuta\n" %x[w])
print("Vrednosti sistema u %f minutu su TT:%f i BI:%f" %(x[w],y[w],z[w]))
f.write("Vrednosti sistema u %s minutu su TT:%s i BI:%s\n" %(x[w],y[w],z[w]))

f.close()
```

3.3.2. Алгоритам Њутнове интерполације

```
# Unos pocetnih vrednosti merenja, njihovo dodavanje u niz i ispis na terminal
# Unos vrednosti krajnje TT
import math

x=[]
y=[]
z=[]

print("================================================================")
n=int(input("Unesi broj pocetnih merenja: "))
print("Unesi vrednosti merenja TT i BB iz tabele:")
for i in range(0,n):
    xtemp=float(input("vreme:"))
    x.append(xtemp)
    ytemp=float(input("TT:"))
    y.append(ytemp)
    ztemp=float(input("BI:"))
    z.append(ztemp)

print("================================================================")
print("Uneta tabela je:")
print("vreme   TT     BI")
for i in range(0,n):
    print(x[i],y[i],z[i])
print("================================================================")
tt=float(input("Unesi vrednost krajnje TT: "))


# Njutnov interpolacioni polinom za vreme-TT
m=n
y_ind=n
xx=x[m-1]+15

while (y_niz[y_ind-1]>=tt):
    x_niz.append(xx)
    print(xx)

    h=x[1]-x[0]

    d=20*[20*[0]]
    for i in range(0,n-1):
        d[i][1]=y[i+1]-y[i]

    ord=4

    for j in range(2,ord):
        for i in range(0,n-j):
            d[i][j]=d[i+1][j-1]-d[i][j-1]

    ind=0
    for i in range(0,n):
        if (x[i]<=xx): ind+=1
```

```
    ind-=1
    p=(xx-x[ind])/h
    yy=y[ind]

    nr=1
    float(nr)
    dr=1
    float(dr)

    for k in range(1,ord):
        nr*=p-k+1
        dr*=k
        yy+=(nr/dr)*d[ind][k]
```

```
    y_niz.append(yy)
    y_ind+=1
    m+=1
    xx+=15
    print(yy)
    yy=0



# Njutnov interpolacioni polinom za vreme-BI

xx=x_niz[0]
#print(xx)
for q in range(0,y_ind):

    h=x[1]-x[0]

    d=20*[20*[0]]
    for i in range(0,n-1):
        d[i][1]=z[i+1]-z[i]

    ord=4

    for j in range(2,ord):
        for i in range(0,n-j):
            d[i][j]=d[i+1][j-1]-d[i][j-1]

    ind=0
    for i in range(0,n):
        if (x[i]<=xx): ind+=1


    ind-=1
    p=(xx-x[ind])/h
    zz=z[ind]
```

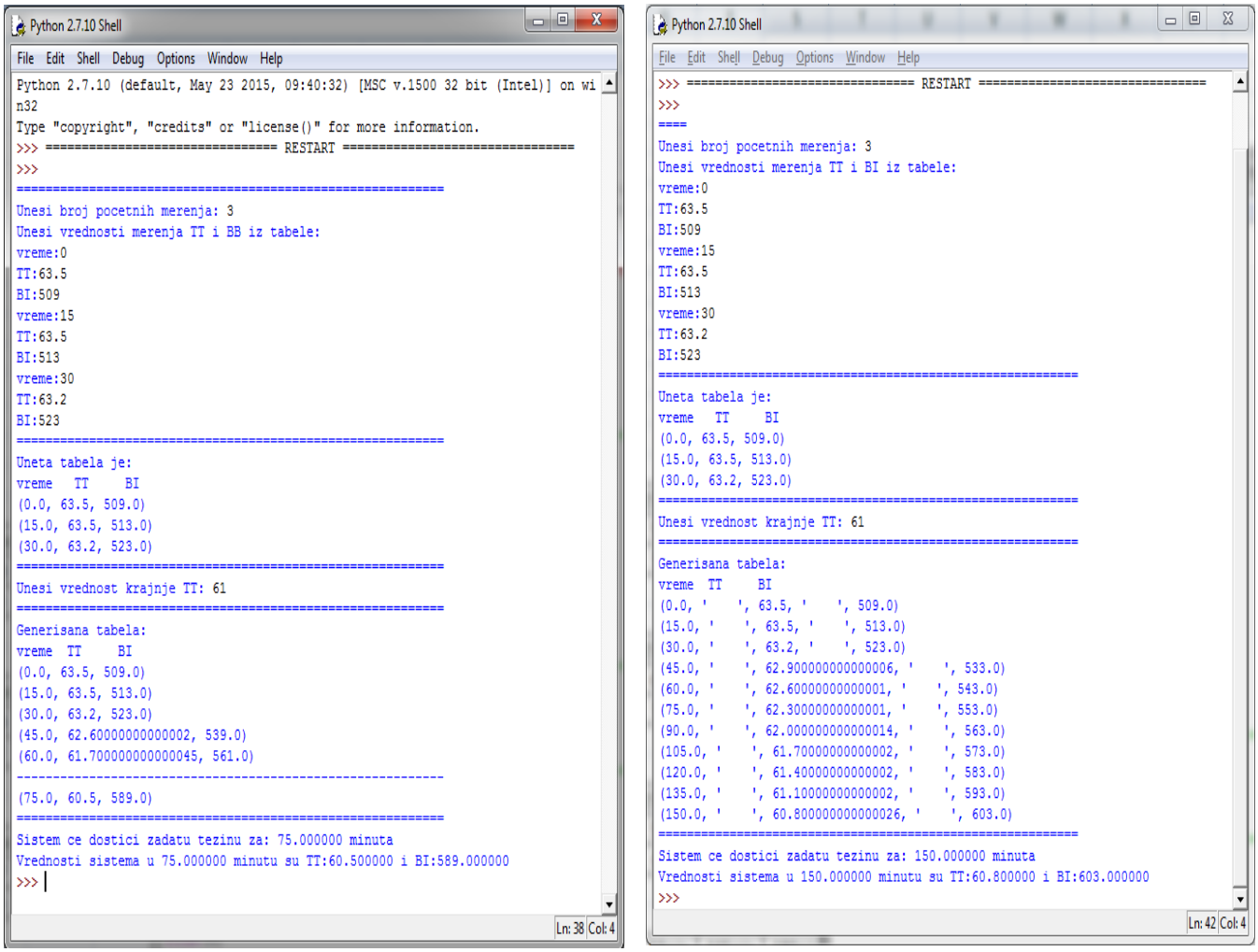
```
 nr=1
 float(nr)
 dr=1
 float(dr)

 for k in range(1,ord):
     nr*=p-k+1
     dr*=k
     zz+=(nr/dr)*d[ind][k]

 z_niz.append(zz)

 xx+=15
 print(zz)
 zz=0



#Ispis generisane tabele i rezultata programa na terminal i u datoteku
f=open("Datoteka.txt","w")

print("================================================================")
print("Generisana tabela:")
f.write("================================================================\n")
f.write("Generisana tabela:\n")


print("vreme  TT     BI")
f.write("vreme  TT     BI\n")
for i in range(0,m+1):
    print(x[i],y[i],z[i])
    f.write("%s   %s   %s\n" %(x[i],y[i],z[i]))
    if (i==m-1):
        print("----------------------------------------------------------------")
        f.write("----------------------------------------------------------------\n")
f.write("================================================================\n")
print("================================================================")
print("Sistem ce dostici zadatu tezinu za: %f minuta" %x[w])
f.write("Sistem ce dostici zadatu tezinu za: %s minuta\n" %x[w])
print("Vrednosti sistema u %f minutu su TT:%f i BI:%f" %(x[w],y[w],z[w]))
f.write("Vrednosti sistema u %s minutu su TT:%s i BI:%s\n" %(x[w],y[w],z[w]))

f.close()
```

3.4. Приказ и провера добијених резултата

У овом поглављу, биће приказани примери рада алгоритма, исписивање излазних вредности, њихов графички приказ као и упоређивање са вредностима мерења процеса дијализе. Примери рада и прикази резултата програма приказани су на следећим сликама.

5. avg	vreme	uključenje	15min	30min	45min	60min	75min	90min	105min	
	TT	63.5	63.5	63.2	62.9	62.7	62.4	62.2	62	
	BI	509	513	523	533	538	546	555	560	
	vreme	120min	135min	150min	**165min**	180min	195min	210min	225min	240min
	TT	61.7	61.5	61.2	**61**	60.7	60.4	60.1	59.9	59.7
	BI	565	569	581	**594**	593	600	608	618	626

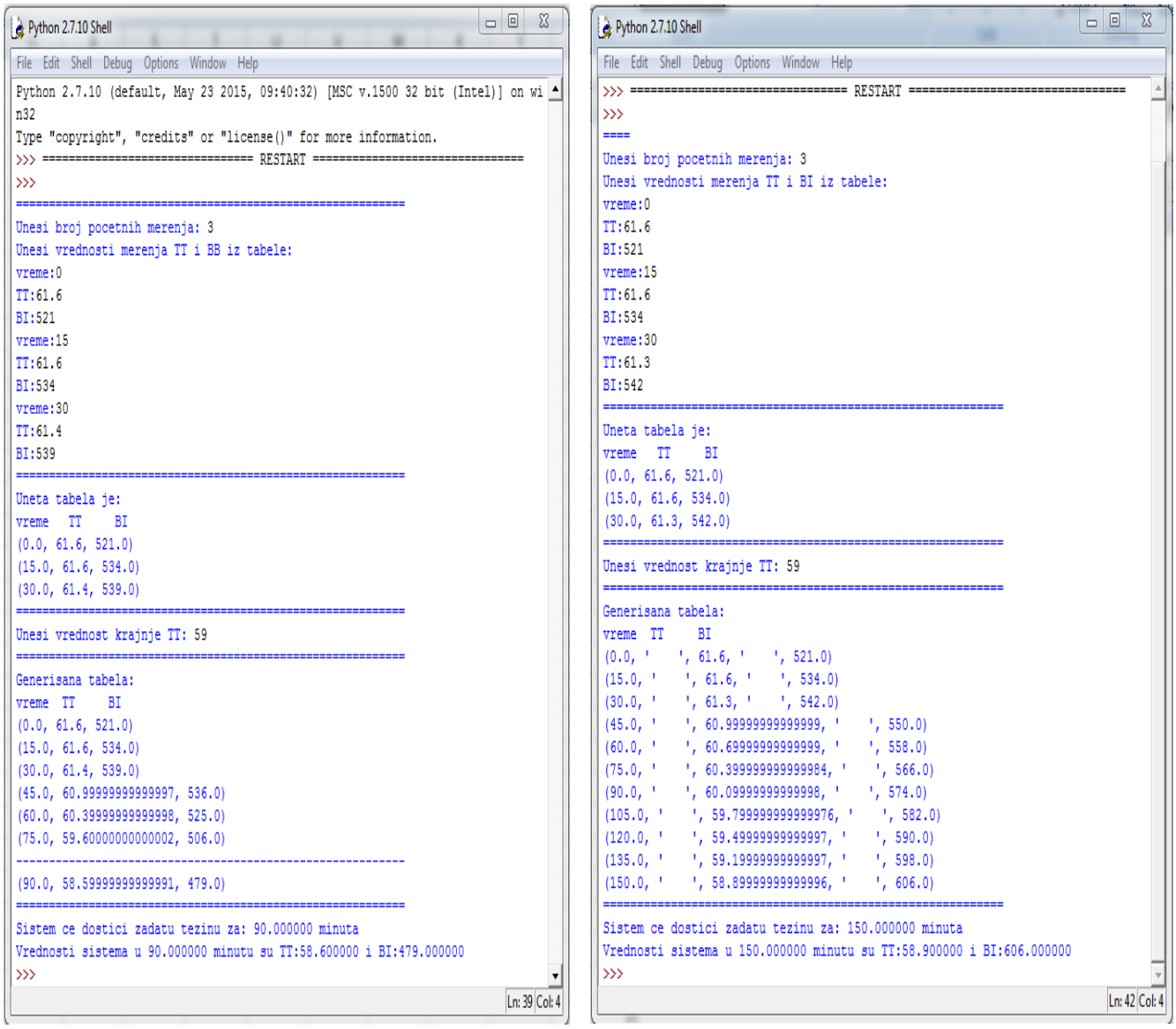
Табела 2. Реалне вредности мерења процеса дијализе, 5. август



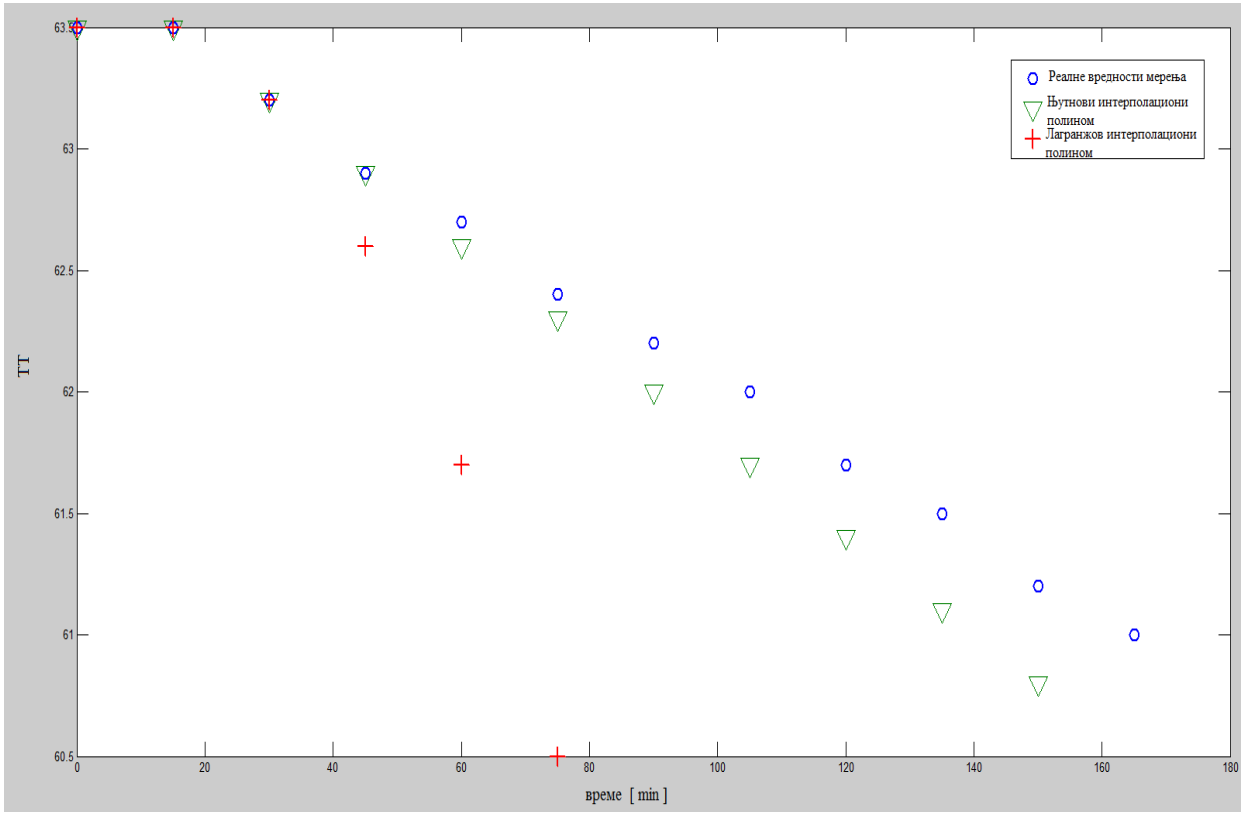
Слика 6. Приказ рада програма за резултате 5. август, Лагранж лево, Њутн десно

7. avg	vreme	uključenje	15min	30min	45min	60min	75min	90min	105min	
	TT	61.6	61.6	61.4	61.3	61	61	60.7	60.4	
	BI	521	534	539	542	555	565	559	563	
	vreme	120min	135min	150min	165min	180min	**195min**	210min	225min	240min
	TT	60.2	60	59.8	59.6	59.4	**59.1**	58.9	58.7	58.5
	BI	571	577	583	592	598	**605**	622	610	621

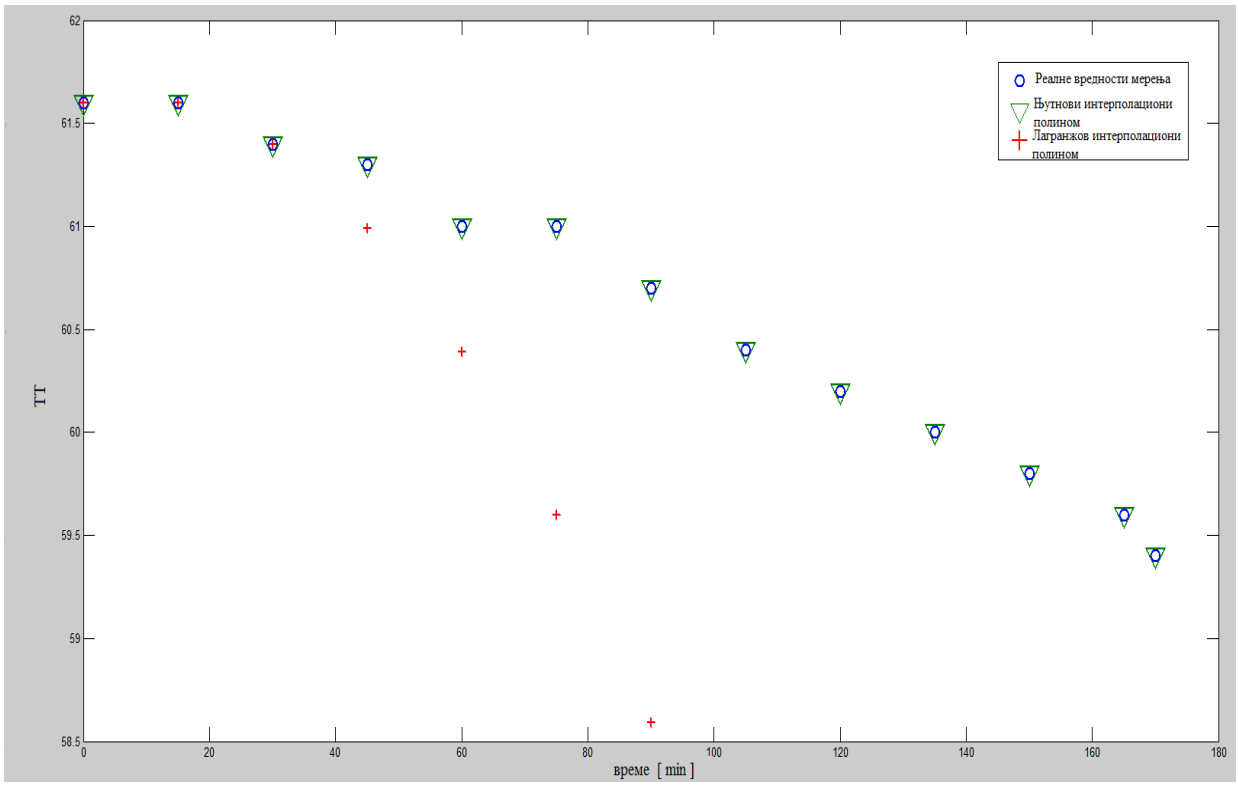
Табела 3. Реалне вредности мерења процеса дијализе, 7. август



Слика 7. Приказ рада програма за резултате 7. август, Лагранж лево, Њутн десно



Слика 7. Графички приказ резултата програма за мерења 5. август



Слика 8. Графички приказ резултата програма за мерења 7. август

4. Закључак

Идентификација и предикција различитих параметара као и креирање математичког модела одређених система широко се користе у различитим врстама истраживања.
Циљ овог рада је реализација аутоматског система за процену параметара током процеса дијализе и предикцију времена трајања процеса. Сам систем је симулиран уз помоћ микрорачунара *Raspberry Pi.*

Како задатак апроксимације представља одређивање функције тако да добијена апроксимативна крива интерполира почетну у одређеним тачкама, у раду смо уз помоћ нелинеарних интерполационих полинома извршили предикцију параметара система.
Од многобројних приступа интерполацији, реализовани су Лагранжов и Њутнов облик полинома у виду писања алгоритма у програмском језику *Python.*

Сама апроксимација параметара овог система подразумевала је праћење два параметра карактеристична за процес дијализе: телесну тежину(ТТ) и индекс тежине(БИ). Уз помоћ поменутих интерполационих метода, за унете почетне вредности реалних мерења, прави се предикција за које време ће систем достићи задату одређену тежину.
Кроз тестирање и анализирање апроксимираних вредности параметара процеса упоређивани су квалитет и тачност коришћених метода чиме су потврђене математичке претпоставке и верификована примена Лагранжовог и Њутновог полинома. Потврђено је да при коришћењу ових облика нелинеарних полинома при степенима вишим од 4, у неким случајевима 5, грешке апроксимације повећавају. Такође потврђено је да је Њутнов приступ у пракси доста прецизнији у поређењу са Лагранжовим.

Захваљујемо се Проф. др Мирјани Костић са Медицинског факултета у Београду, која је корисним сугестијама и уступљеним реалним измереним вредностима помогла реализацији овог рада.

5. Литература

[1] Mirko Vujosevic, *Identifikacija i Aproksimacija*, Beograd, 2013.

[2] Liming Dai, Reza N. Jazar, *Nonlinear Approaches in Engineering Application*, Springer New York Dordrecht Heidelberg, 2012.

[3] Wilson J. Rugh, *Nonlinear System Theory*, The Johns Hopkins University, 2002.

[4] Ronald A. DeVore, *Nonlinear approximation and its applications*, Springer Texas A&M University, 2009.

[5] Dogan Ibrahim, *Raspberry Pi Hardverski Projekti*, Infoelektronika, 2014.

[6] Simon Monk, *Raspberry Pi Cookbook*, O'REILLY, 2014.

[7] Body Impedance Analyzer, Darmstadt, Germany

[8] Peter Kotanko, Nathan W. Levin, Fansan Zhu, *Current state bioimpedance technologies in dialysis*, Nephrology Dialysis Transplantation, 2008.

[9] Franz Schaefer, Elke Wuhl, Reinhard Feneberg, Otto Mehls, Karl Scharer, *Assessment of body composition in children with chronic renal failure*, IPNA, 2000.

[10] Wiring Pi - GPIO Interface library for the Raspberry Pi, *http://wiringpi.com/download-and-install/* приступљено 17.11.2015