TARTU ÜLIKOOL

MATEMAATIKA-INFORMAATIKATEADUSKOND

Arvutiteaduse instituut

Informaatika õppekava

Maarja Lepamets

PCR-i praimerite kvaliteedimudeli integreerimine Primer3-e disainimetoodikasse

Magistritöö (30 EAP)

Juhendaja: Lauris Kaplinski, PhD

Tartu 2015

PCR-i praimerite kvaliteedimudeli integreerimine Primer3-e disainimetoodikasse

Lühikokkuvõte:
Polümeraasi ahelreaktsiooni (PCR) edukas toimumine sõltub praimerite lähtejärjestusega seondumise stabiilsusest ja spetsiifilisusest. Praimeridisaini protsessis tuleb seega jälgida, et disainitaval praimeril ei oleks uuritavas genoomis palju mittespetsiifilisi seondumiskohti. Käesoleva töö raames koostati mudel, mis hindaks praimeri mittetöötamise tõenäosust, lähtudes selle potentsiaalsete seondumiskohtade arvust uuritavas genoomis. Seejärel implementeeriti maskeerimisalgoritm, mis kasutaks loodud mudelit lähtejärjestusest ebasobivate praimeri seondumiskohtade maskeerimiseks. Kirjeldatud algoritm integreeriti praimeridisaini programmi Primer3. Maskeerimisfunktsiooni kasutamine praimeridisaini kontekstis vähendas disainitavate praimerite keskmist mittetöötamise tõenäosust enam kui kaks korda.

Võtmesõnad: PCR, Primer3, praimeridisain, k-meerid, järjestuse maskeerimine

The integration of quality model to the primer design process with Primer3 software

Abstract:
The success rate of polymerase chain reaction (PCR) is highly dependent on the binding specificity and stability of the primers currently in use. Primers with more binding sites tend to have higher failure rates. In the current study a model for predicting primer failure rates based on the number of their binding sites was fit. Next, the model was used in an algorithm which would mask the regions of a given input sequence that are unsuitable for primer design. The described algorithm was then integrated to Primer3 software where it was tested in the context of primer design. Our results show that by using the masking functionality one can reduce the primer failure rate more than two times.

Keywords: PCR, Primer3, primer design, k-mers, sequence masking

Sisukord

Sissejuhatus **5**

1 Kirjanduse ülevaade **6**

1.1 Praimerite kasutamine ja disainimine 6

1.1.1 Polümeraasi ahelreaktsioon 6

1.1.2 Praimeridisain 7

1.1.3 Seondumiskohtade negatiivne mõju praimerite töökindlusele . . . 9

1.1.4 Primer3 10

1.2 K-meeride sagedustel põhinev järjestuse maskeerimine 13

2 Töö eesmärgid **16**

3 Metoodika ja implementatsioon **17**

3.1 Praimerite andmestik 17

3.2 Praimeri mittetöötamist hindava mudeli koostamine 18

3.2.1 Praimeri mittetöötamise kirjeldamine logistilise regressiooni abil . 18

3.2.2 Tunnuste valimine mudelisse 19

3.2.3 Bioloogiliselt relevantse vabaliikme leidmine 22

3.3 Maskeerimisalgoritm 22

3.3.1 Sisend ja väljund 23

3.3.2 Ringpuhver 24

3.3.3 Järjestuse maskeerimine 25

3.4 Maskeerimisfunktsionaalsuse integreerimine Primer3 programmi 27

4 Tulemused **30**

4.1 Maskeerimisalgoritmi funktsionaalsus 30

4.2 Maskeerimisalgoritmi mudel 31

4.3 Maskeerimisalgoritmi aja- ja mälukasutus 35

4.4 Maskeerimisalgoritmi kasutamine Primer3-e koosseisus 37

5 Arutelu **38**

Kokkuvõte **41**

Kirjanduse loetelu **42**

Lisad **46**

Lisa A: Praimerite andmestiku näidis 46

Lisa B: AIC-l põhineva algoritmiga genereeritud mudelid 47
Lisa C: Maskeerimisprogrammi käsureaparameetrid 48
Lisa D: Maskeerimisprogrammi käsurea näited 49
Lisa E: Primer3-e sisendi ja väljundi näited 50

Lihtlitsents **51**

Sissejuhatus

Polümeraasi ahelreaktsioon (PCR) on meetod, mis võimaldab lihtsalt ja kiirelt laboritingimustes uuritavat DNA lõiku paljundada. Seda kasutatakse paljudes praktilistes rakendustes, näiteks kliinilises diagnostikas, isikutuvastamisel ja mujal. PCR on tsükliline protsess. Iga tsükli alguses toimub lühikeste DNA järjestuste – praimerite – seondumine paljundatavale lähte-DNA-le. Nendest seondumispiirkondadest algab uute DNA ahelate süntees. Seetõttu sõltub PCR-i õnnestumine suuresti sellest, kui spetsiifiliselt ja stabiilselt praimerid seonduvad. Mittespetsiifiline praimerite seondumine võib viia valeproduktide tekkeni.

Tagamaks praimerite eduka töötamise, tuleb nende järjestuste disainimisel järgida mitmeid kriteeriumeid. Disainiprotsessi lihtsustab selleks ettenähtud programmide kasutamine. Primer3 on sellistest programmidest üks populaarsemaid. See võtab sisendiks huvipakkuva järjestuse ning disainib lähtejärjestusele seonduvad praimerid, arvestades peaaegu kõigi teadaolevate valikukriteeriumitega. Paraku ei ole seni Primer3-e algoritmi lisatud funktsionaalsust, mis võimaldaks kandidaatpraimereid välja selekteerida nende mittespetsiifiliste seondumiskohtade arvu alusel.

Käesoleva magistritöö eesmärkideks on koostada mudel praimerite mittetöötamise tõenäosuse hindamiseks, lähtudes nende potentsiaalsete seondumiskohtade arvust, ning seejärel implementeerida maskeerimisalgoritm, mis kasutaks leitud mudelit selliste järjestuse piirkondade maskeerimiseks, millele disainitavate praimerite mittetöötamise tõenäosus on kasutaja määratud nivoost suurem. Töö lõppeesmärgiks on nimetatud maskeerimisfunktsionaalsuse integreerimine Primer3 programmi.

Töö koosneb kirjanduse ülevaatest ja praktilisest osast. Kirjanduse ülevaates antakse ülevaade PCR-i metoodikast ja praimeridisainist ning tutvustatakse lähemalt programmi Primer3. Lisaks räägitakse lühidalt k-meeride loendamise programmist GenomeTester4 ja sellest, kuidas on k-meeride sagedustabeleid järjestuse maskeerimiseks võimalik kasutada. Praktilises osas kirjeldatakse praimerite mittetöötamise tõenäosust hindava mudeli koostamise algoritmi ja implementeeritud maskeerimise programmi loogikat. Eraldi räägitakse maskeerimisalgoritmi integreerimisest Primer3 programmi. Töö viimases osas esitatakse tulemused ja arutelu.

Kogu metoodika peatükis kirjeldatud tööprotsess on läbi viinud käesoleva töö autor. Erandiks on mudeli loomisel kasutatud praimerite disainimine ja testimine laboritingimustes. Antud magistritöö on osaliselt järg Tartu Ülikooli molekulaar- ja rakubioloogia instituudis 2014. aastal kaitstud bakalaureusetööle „Oligomeeridel põhinevate bioinformaatiliste meetodite kasutamine bakterite määramiseks sekveneerimislugemitest", mille käigus loodi k-meeride lugemise ja analüüsimise pakett GenomeTester4.

1 Kirjanduse ülevaade

1.1 Praimerite kasutamine ja disainimine

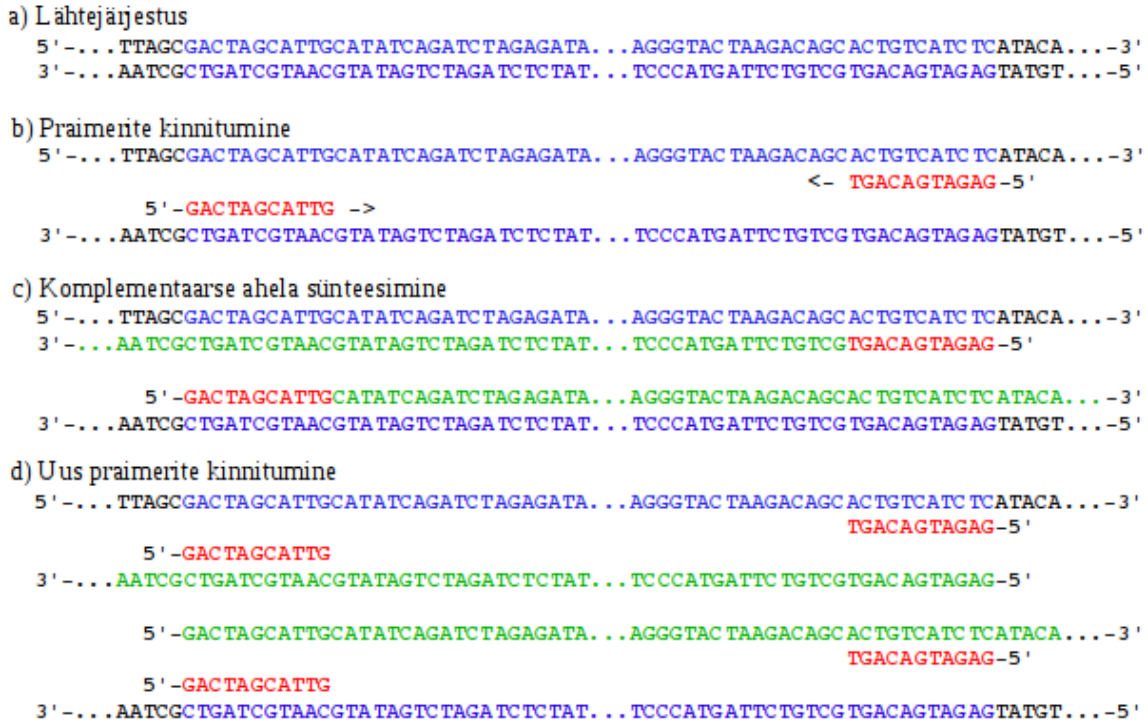
1.1.1 Polümeraasi ahelreaktsioon

Polümeraasi ahelreaktsioon (PCR, ingl. k. *polymerase chain reaction*) (Mullis *et al.*, 1984) on biotehnoloogias ja molekulaarbioloogias üks enim kasutatavaid meetodeid DNA järjestuse amplifitseerimiseks ehk paljundamiseks. PCR-i abil saab luua mingist kindla järjestusega DNA piirkonnast kiiresti miljoneid koopiaid. Tänu protsessi selektiivsetele omadustele saab nii ka kindlaks teha, kas uuritav DNA sisaldab mingit konkreetset järjestust, samuti määrata selle järjestuse pikkust ning korduste arvu. Reaalajaline kvantitatiivne PCR (qPCR) võimaldab mõõta huvipakkuvate DNA või RNA järjestuste, näiteks transkriptide, koopiate arvu algses reaktsioonisegus. PCR leiab praktilist rakendust näiteks meditsiinilises diagnostikas (Valones *et al.*, 2009), isikutuvastamisel (Thompson *et al.*, 2012) ning mujal. PCR on kasutusel ka kõigi laialt kasutatavate sekveneerimistehnoloogiate esimestes etappides DNA fragmentide paljundamiseks.

PCR-i tööpõhimõte on rajatud sellele, et DNA on harilikes tingimustes kaheahelaline, kusjuures ahelad on teineteise suhtes antikomplementaarsed (paarikaupa paarduvad nukleotiidid A ja T ning C ja G). Pärast ahelate lahutamist on antikomplementaarsust ära kasutades võimalik kummagi ahela järgi sünteesida teise ahela järjestus. Elusorganismide rakkudes on üheahelalisele DNA-le temaga komplementaarse ahela sünteesimiseks ensüüm DNA polümeraas. Seda kasutatakse rakkude paljunemisel geneetilise koodi duplitseerimiseks järglasrakkudesse (Stratmann ja van Oijen, 2014).

PCR-i reaktsioon toimub lahuses, mille põhilised koostisosad on lisaks kuumaresistentsele polümeraasile ning paljundatavale lähte-DNA järjestusele veel kaks praimerijärjestust (kumbagi miljoneid koopiad), vabad nukleotiidimolekulid, puhverlahus reaktsiooni pH säilitamiseks ja magneesiumi- ning kaaliumiioonid DNA polümeraasi töö stabiliseerimiseks (Kramer ja Coen, 2001). Praimerid on lühikesed, enamasti 16 – 28 nukleotiidi pikkused üheahelalised DNA fragmendid (Wu *et al.*, 2004), mis paarduvad antikomplementaarselt paljundatava DNA piirkonna mõlema otsa erinevate ahelatega. Praimeri ja üheahelaliseks muudetud lähte-DNA paardumisel tekib kaheahelaline fragment, mille otsa seondub polümeraas ja hakkab DNA-d täies pikkuses kaheahelaliseks sünteesima.

PCR on tsükliline protsess, mille iga kordusega suurendatakse lahuses olevate paljundatava DNA lõikude arvu ligikaudu kaks korda. Üks tsükkel koosneb kolmest faasist: kaheahelalise DNA ahelate lahutamine üheahelalisteks, praimerite seondumine üheahelalise DNA otstesse ja komplementaarse ahela sünteesimine (vt joonis 1). Faaside vaheldumine teostatakse reaktsioonikeskkonna temperatuuri muutmisega.



Joonis 1: PCR-i tööpõhimõte: *PCR-i kasutatakse etteantud järjestuse mingi piirkonna (tähistatud sinisega) paljundamiseks (a). Esmalt lahutatakse kaheahelaline DNA üheahelalisteks. Seejärel seonduvad praimerid (tähistatud punasega) paljundatava piirkonna kumbagi otsa (b). Kinnitunud praimeritelt algab lähtejärjestusega komplementaarse ahela (tähistatud rohelisega) süntees suunaga 5'→3' (c). Sünteesi lõppedes ahelad lahutatakse, uued praimerid saavad seonduda ja tsükkel hakkab otsast peale (d).*

1.1.2 Praimeridisain

PCR-i õnnestumist mõjutavad mitmed tegurid. Üheks olulisemaks neist on praimerite korrektne seondumine paljundatava DNA järjestusele. Reaktsiooni edukaks läbiviimiseks on vaja, et praimerid kinnituks piisavalt stabiilselt ja samas väga spetsiifiliselt ainult ettenähtud kohtadele ega moodustaks omavahelisi struktuure. Suurema praimerite töökindluse saavutamiseks on teadlased kindlaks teinud mitmeid nende järjestusi puudutavaid kriteeriumeid, mida praimerite valikul arvestada. Biotehnoloogias nimetatakse sobivate praimerite järjestuste leidmist praimeridisainiks.

PCR-i praimeripaar disainitakse selliselt, et kumbki praimer paarduks ühe paljundatava DNA piirkonna otsaga. See tähendab, et praimeri ja lähte-DNA otsa järjestused peavad olema omavahel antikomplementaarsed. PCR-i praimerid peavad olema pikkusega 16 – 28 nukleotiidi, kusjuures kahe praimeri pikkuste erinevus ei tohiks olla suurem kui 3 nukleotiidi, ja eelistatult GC-sisaldusega vahemikus 40 – 60% (Wu *et al.*, 2004).

Praimereid ei disainita genoomis kordusjärjestustele ega mõndade teiste genoomipiirkondadega suurt homoloogiat omavatele aladele. Uuritavas genoomis mitmes sarnases koopias olevale järjestusele disainitud praimer võib PCR-i käigus valesse kohta seonduda ja põhjustada valeprodukti tekke. Praimer võib paarduda ka genoomipiirkonnaga, mis pole sellega ideaalselt antikomplementaarne, kuid on antikomplementaarsele järjestusele piisavalt sarnane (näiteks sisaldab ühenukleotiidset variatsiooni). Taoliselt kinnitunud praimerilt võib DNA süntees samuti alata, kuid tihti toimub see väiksemas ulatuses kui ideaalselt paardunud praimerilt alanud DNA süntees (Ghedira *et al.*, 2009; Sipos *et al.*, 2007). Mitte-ideaalset paardumist lubatakse eelkõige juhul, kui on vaja disainida mitmele järjestusele universaalseid praimereid. Üldjuhul valitakse praimerid nii, et nende seondumiskohad oleksid võimalikult spetsiifilised.

Lisaks mittespetsiifilisele paardumisele lähte-DNA-ga võivad valesti valitud praimerid paarduda ka omavahel. Sellist struktuuri nimetatakse praimerite dimeeriks ning see tekib juhul, kui ühe praimeri järjestus on osaliselt antikomplementaarne teise praimeri järjestusega. Oluline on ka jälgida, et praimer ei sisaldaks pööratud kordusjärjestust. Vastasel korral võib praimer paarduda iseendaga ja moodustada n-ö juuksenõela struktuuri. Praimerid, mis on reaktsioonisegus dimeeri või juuksenõela struktuurina, ei paardu enam lähte-DNA-ga ega osale seega oodatava produkti amplifikatsioonil. (Vallone ja Butler, 2004)

Sulamistemperatuuriks (T_m) nimetatakse praimeridisaini kontekstis temperatuuri, millest madalamate temperatuuride juures on praimer eelistatult paardunud olekus ja kõrgematel temperatuuridel eelistatult vabalt lahuses. T_m sõltub praimeri pikkusest ja nukleotiidsest koostisest. Pikematel ja GC-rikkamatel praimeritel on kõrgem T_m. Praimerite T_m arvutamiseks kasutatakse enamasti SantaLucia ja Hicks (2004) termodünaamika mudeleid, mis arvestavad lisaks iga praimerijärjestuses oleva nukleotiidi paardumise energeetilisele efektile ka samal ahelal teineteise vahetus naabruses olevate nukleotiidide omavaheliste interaktsioonide (ingl. k. *stacking*) mõju. PCR-i praimerite sulamistemperatuur on harilikult vahemikus 50 – 62°C (Chuang *et al.*, 2013). Praimerite seondumise faasis kasutatakse sellest mõnevõrra madalamat temperatuuri, et soodustada praimerite paardumist lähte-DNA-le. Liiga madalatel temperatuuridel võib aga praimer seonduda mittespetsiifiliselt, mis takistab reaktsiooni toimumist või põhjustab valeprodukti tekkimist. Praimeridisainil tuleb jälgida, et praimeripaari kasutades oleks mõlema praimeri T_m enam-vähem võrdne.

Lineaarse DNA molekuli otsad ei ole keemiliselt samaväärsed ja neid nimetatakse 5'- ja 3'-otsadeks. DNA ahela järjestust loetakse suunal 5'→3'. Kaheahelalise DNA ahelad on teineteise suhtes vastupidise suunaga. Polümeraas kinnitub praimeri 3'-otsa ning liigub lähte-DNA-l edasi suunal 3'→5', sünteesides uut komplementaarset ahelat suunaga 5'→3'

(vt joonis 1). Tagamaks polümeraasi edukat kinnitumist praimeri seondumiskohta, peab praimeri 3'-ots olema lähte-DNA-le seondunud võimalikult stabiilselt. Sageli soovitatakse, et praimeri 3'-otsas paikseks G või C nukleotiid (Wu *et al.*, 2004), sest nende omavaheline paardumine on tugevam kui A ja T nukleotiididel (moodustavad rohkem vesiniksidemeid). Samas tuleks vältida GC kõrvutiasumist, sest see võib põhjustada sekundaarstuktuuride teket 3'-otsas (Onodera ja Melcher, 2004). Kui ollakse sunnitud kasutama praimerit, mis pole seondumiskohaga täies ulatuses antikomplementaarne, tuleks eelistada juhtu, kus mittekomplementaarsed aluspaarid ei asuks praimeri 3'-otsa lähedal.

Praimerite disainimiseks on loodud mitmeid programme, mis arvestavad oma disainiprotsessis kõigi ülalmainitud tingimustega. Nendest tuntumad on näiteks OLIGO (Rychlik, 2007), PrimerPremier (PREMIER Biosoft), FastPCR (PrimerDigital) ja Primer3 (Rozen ja Skaletsky, 2000). Võimalike mittespetsiifiliste seondumiskohtade leidmiseks joondatakse disainitud praimereid BLAST (Altschul *et al.*, 1997), BLAT (Kent, 2002), SSAHA (Ning *et al.*, 2001) või muude sarnaste programmidega uuritava organismi genoomile. Lisaks on hulk programme, mis võtavad sisendiks lähte-DNA ja praimerite järjestused ning ennustavad PCR-il tekkivaid produkte. Seda nimetatakse digitaalseks või elektrooniliseks PCR-iks (ePCR). Digitaalse PCR-i tarkvarad on näiteks In-Silico PCR (Kent, UCSC) ja Reverse ePCR (http://www.ncbi.nlm.nih.gov/projects/e-pcr/reverse.cgi). Käesolev töö on keskendunud eelkõige PCR-i praimeripaaride disainimisele Primer3 programmiga.

1.1.3 Seondumiskohtade negatiivne mõju praimerite töökindlusele

Praimerite mittespetsiifilised seondumised mõjutavad PCR-i kvaliteeti (Ghedira *et al.*, 2009). Andreson *et al.* (2008) on oma töös koostanud statistilised mudelid PCR-i ebaõnnestumise ennustamiseks praimerite võimalike seondumiskohtade arvu järgi inimese genoomis. Mudelite loomise aluseks on võetud 236 tunnust, millest enamik vastab vaadeldava praimeri 3'-otsa või sellega sarnaste järjestuste sagedustele inimese genoomis. Lisaks on tunnuste sekka kaasatud ka praimeri pikkus, GC-protsent, sulamistemperatuur ning muud seondumiskohtade arve mittepuudutavad praimerite omadused.

Lõplike statistiliste mudelite koostamisel on silmas peetud, et need kirjeldaks võimalikult väheste tunnuste abil võimalikult suurt hulka uuritava tunnuse koguvarieeruvusest. Andreson *et al.* (2008) töö käigus on loodud viis erineva kompleksusega üldistatud lineaarset mudelit. Neist kaks (GM1 ja GM1MM) on toodud käesoleva töö peatükis 4.2 tabelis 2 (nimetatud on ainult seondumiskohtade arve puudutavad mudeli tunnused).

Loodud lineaarsetest mudelitest nelja saab kasutada lähte-DNA järjestuse nende piirkondade maskeerimiseks, millele disainitud praimeri mittetöötamise tõenäosus on suur. Praimeridisainil nimetatakse maskeerimiseks lähte-DNA järjestuses tekstikujul esi-

tatud nukleotiidisümbolite (ACGT) asendamist spetsiaalse sümboliga (tavaliselt N või väiketäht), mis on programmile märgiks, et sellesse kohta ei tohi praimereid disainida.

Parim loodud mudelitest (GM1) aitab ebaõnnestumise tõenäosust vähendada ligikaudu 3 korda (uus keskmine ebaõnnestumise tõenäosus on varasema 17% asemel 6%). See mudel sisaldab nelja tunnust: praimeri 3'-otsas olevate 16- ja 14-meeride seondumiskohtade arve, 15-meeride seondumiskohtade arvude ruutu ning praimeri 3'-otsas oleva 16-meeri GC-protsenti. Paremuselt järgmine mudel (GM1MM) sisaldab tunnustena ka praimeri selliste seondumiskohtade arve, mis pole praimerijärjestusega täies ulatuses antikomlementaarsed. See mudel annab mõnedes olukordades paremaid hinnanguid, kuid selle arvutamine võtab esimese mudeliga võrreldes palju rohkem aega. Ülejäänud kaks mudelit neljast on termodünaamikapõhised ning nende arvutamine on eelmistest mudelitest tunduvalt keerulisem. Viies mudel hindab PCR-i ebaõnnestumise tõenäosust, lähtudes mõlema praimeri ja PCR-i produkti terviklikest järjestustest. (Andreson *et al.*, 2008)

1.1.4 Primer3

Primer3 (Rozen ja Skaletsky, 2000) on üks populaarsemaid praimeridisaini programme. Hetkeseisuga (03.11.2015) on selle viimane versioon 2.3.6. Primer3-e pakett koosneb käsurealt käivitatavast alusprogrammist ja kahest veebiliidesest: Primer3Web ja Primer3Plus (Untergasser *et al.*, 2007). Primer3-e käsurea versioon on eelkõige mõeldud integreerimiseks teiste programmidega ja suure arvu praimerite disainiks. Veebiliidesed on mõeldud tavakasutajatele, kelleks enamasti on laboris või looduses töötavad bioloogid, kes ei disaini ühe analüüsi käigus korraga suurt hulka praimereid. Primer3-e esimene versioon avaldati 2000. aastal (Rozen ja Skaletsky, 2000) ja see loodi varasema programmi Primer 0.5 (Daly *et al.*, 1991) uue implementatsioonina.

Primer3 on vabavara. Alusprogrammi kood on kirjutatud programmeerimiskeeles C ning on käivitatav käsurealt. Programmi disainil on arvestatud, et seda oleks lihtne kaasata mitmeid programme sisaldavatesse bioinformaatilistesse tööjärjekordadesse või integreerida teiste tarkvarapakettide sisse. On loodud mitmeid programme ja veebipõhiseid teenuseid, mis kasutavad ühe komponendina Primer3-e, näiteks PCRTiler (Gervais *et al.*, 2010), Primer-BLAST (Ye *et al.*, 2012) ning RexPrimer (Piriyapongsa *et al.*, 2009). Integreerimise hõlbustamiseks on koostatud Primer3-e funktsioonidest eraldi C/C++ moodul (Untergasser *et al.*, 2012).

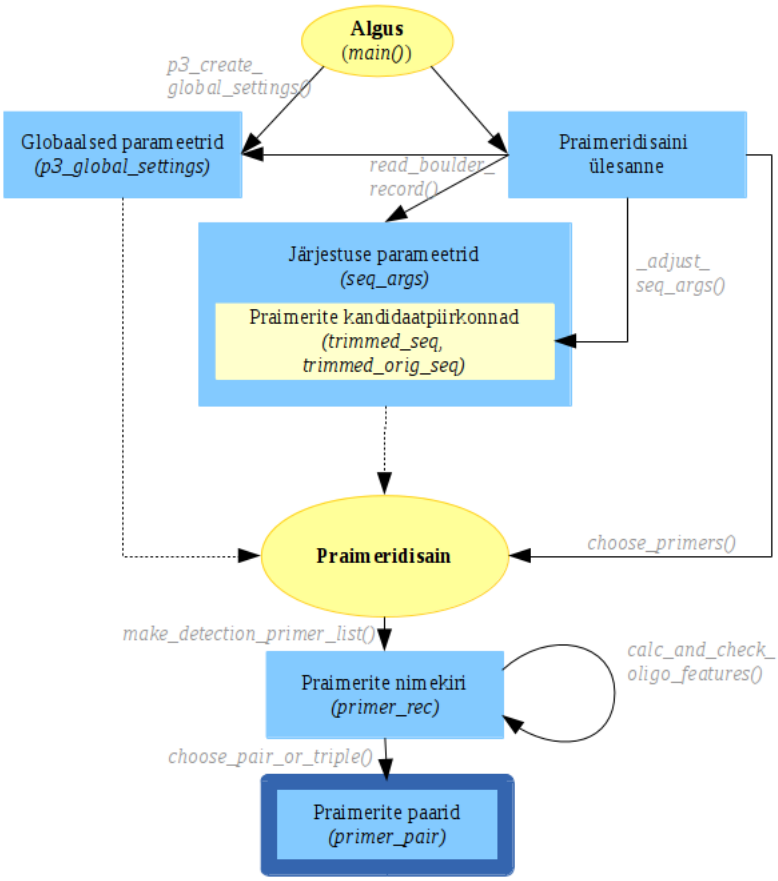
Primer3-e sisendiks on nukleiinhappejärjestus ning kasutaja määratud parameetrite väärtused. Programmi tööülesandeks on PCR-i praimerite, sekveneerimispraimerite ja/või hübridisatsiooniproovide disainimine antud järjestuse paljundamiseks, analüüsiks või detekteerimiseks. Samuti võimaldab Primer3 kontrollida juba olemasolevate praimerijärjestuste sobivust etteantud nukleotiidse järjestuse paljundamiseks.

Kasutaja saab Primer3-le ette anda mitmeid parameetreid. Nii saab programmi kasutada (sõltuvalt püstitatud ülesandest) väga erinevate omadustega praimerite disainimiseks. Kõige tähtsamad parameetrid on lisaks lähtejärjestusele veel oodatavate praimerite pikkuste, sulamistemperatuuride ja GC-protsentide lubatud vahemikud. Kasutaja saab programmile ette anda ka selliste järjestuste andmebaasi, millele seonduvaid praimereid ei soovita (näiteks teadaolevad kordusjärjestused). Olulised parameetrid on veel praimerite 3'-otsa seondumise stabiilsus ning võimalike sekundaarstruktuuride ja praimerite dimeeride stabiilsused. Kokku on programmil enam kui 150 erinevat parameetrit. Primer3-e väljundiks on nimekiri sobivatest praimeritest, praimeripaaridest või hübridisatsiooniproovidest. Need valitakse lähtudes etteantud parameetritest, mida kasutades püüab Primer3 leida sellised praimerid või proovid, mis vastaks kõige paremini kasutaja soovidele. Tihti väljastab programm mitu erinevat kandidaati. Sellisel juhul on need järjestatud kasutaja määratud optimumiga sobivuse aluse. (Rozen ja Skaletsky, 2000; Untergasser *et al.*, 2012)

Mõlemad veebiliidesed – Primer3Web ja Primer3Plus – võimaldavad kasutada alusprogrammi kogu funktsionaalsust. Viimane neist on oma kasutajaliidese poolest mugavam kui Primer3Web, sest parameetrid on süstematiseeritud tavakasutaja vajadustest lähtuvalt ning detailsemad ja vähemkasutatavad parameetrid on vaikimisi varjatud. Samuti on Primer3Plus-iga automatiseeritud mõned mitmeetapilised protsessid, mida ainult alusprogramm ei võimalda teostada, näiteks kloneerimiseks mõeldud praimerite disainimine. Lisaks on Primer3Plus-iga disainitud praimereid võimalik firmadelt tellida veebilehelt lahkumata.

Töö käigus leiab Primer3 esmalt nimekirja praimeritest, mis vastavad kasutaja määratud kriteeriumitele. Saadud nimekiri sorteeritakse ja (PCR-i praimerite disaini puhul) leitakse praimerite paarikaupa sobivused (vt ka joonis 2). Üksikute kandidaatpraimerite ja -praimeripaaride testimine on üles ehitatud selliselt, et esmalt elimineeritakse kiiremate testidega suur osa kandidaatidest ning ajakulukamad operatsioonid nagu sekundaarstruktuuride ja dimeeride stabiilsuste arvutamine jäetakse algoritmi viimasteks sammudeks, kui enamik praimereid on juba analüüsist eemaldatud. Iga praimeri ja praimeripaari kohta talletatakse kvaliteedihinnang, mille alusel väljastatakse kasutajale parimad praimerid. (Untergasser *et al.*, 2012)

Primer3-e alusprogrammile saab kasutaja parameetrite väärtuseid anda tekstifailiga, mis vastab nn boulder IO formaadile (vt näidet lisast E). Selle faili igal real on parameetri nimi (ingl. k. *tag*) ja väärtus eraldatud võrdusmärgiga. Samas formaadis on vaikimisi ka Primer3-e väljundfailid. See võimaldab saadud väljundit kergesti mõne järgmise programmi sisendiks konverteerida. Primer3 võtab sisendiks kahte tüüpi parameetreid. Globaalsete parameetrite (nimed algavad „PRIMER_") väärtused püsivad konstantsete-



Joonis 2: PCR-i praimerite disainimine Primer3 programmiga: *Primer3 programmi käivitamisel väärtustatakse esmalt globaalsed parameetrid (p3_global_settings). Iga praimeridisaini ülesande alguses neid vajadusel uuendatakse. Lisaks väärtustatakse kasutaja määratud järjestuse parameetrid (seq_args). Seejärel leitakse sisendjärjestusest see piirkond, millele praimereid disainitakse (trimmed_seq). Kõiki parameetreid arvesse võttes leitakse esialgne praimerite (primer_rec) nimekiri. Seda nimekirja vähendatakse jooksvalt, eemaldates sealt praimereid, mis kasutaja määratud kriteeriumitega ei sobi. Ülejäänud praimeritele arvutatakse kvaliteediskoorid (calc_and_check_oligo_features()). Allesjäänud praimerite hulgast leitakse sobivad praimeripaarid (primer_pair), mis väljastatakse pingerea alusel.*

na kogu programmi töö vältel või kuni kasutaja neid muudab. Järjestuse parameetrid (nimed algavad „SEQUENCE_") sisaldavad informatsiooni ühe konkreetse praimeridisaini ülesande kohta ja tuleb seega iga uue ülesande korral uuesti väärtustada. Primer3-e hilisemates versioonides saab kasutaja programmile globaalsed parameetrid anda eraldi tekstifailiga (Untergasser *et al.*, 2012). Eelkõige lisab see mugavust Primer3-e veebiliideste kasutamisel.

Pärast esmast avaldamist on Primer3-e programmi mitmel korral täiendatud ja parandatud. Oluliseks muudatuseks on originaalversioonis kasutuselolevate termodünaamika valemite asendamine alates versioonist 1.1 uuemate ja täpsemate vastu, mis kasutavad praimerite sulamistemperatuure ning sekundaarstruktuuride tekkimise tõenäosuseid arvutades (SantaLucia ja Hicks, 2004) lähimnaabri (ingl. k. *nearest-neighbour*) mudeleid ning arvestavad ka PCR-i lahuses olevate soolade mõju praimerite seondumisele. Ühtlasi on täiustatud praimerite disainimist maskeeritud aladega järjestustele. Algsest versioonist peale on Primer3 arvestanud järjestuses N-ga maskeeritud alasid (n-ö *hard-masking*). Selliste aladega ülekattes olevaid praimereid programm vaikimisi ei disaini. Alates versioonist 1.1 võtab Primer3 arvesse ka väiketähtedega maskeeritud järjestust (n-ö *soft-masking*). Selline maskeerimine erineb programmi töö seisukohalt N-ga maskeerimisest selle poolest, et Primer3 lubab väiketähega maskeeritud nukleotiidiga ülekattes praimerit disainida, juhul kui maskeeritud nukleotiid ei asu praimeri 3'-otsas. (Kõressaar ja Remm, 2007)

Muuhulgas on täienduste seas veel parameetreid, millega kasutaja saab nõuda, et iga Primer3-e väljastatud praimeripaar sisaldaks unikaalseid praimereid, mis erineks omavahel rohkem kui etteantud nukleotiidide arvu võrra. Lisaks on hõlbustatud praimeripaaride disainimist ekson-ekson üleminekutele (ingl. k. *splice site*), arvestamata sealjuures nende vahele jäävat intronit. (Untergasser *et al.*, 2012)

Primer3 ei ole mõeldud kõigi spetsiifiliste praimeridisaini probleemide lahendamiseks. Keerulisemate ülesannete jaoks tuleb seda kombineerida teiste programmidega. Funktsionaalsused, mida Primer3 ei paku, kuid mis on saavutatud programmi kasutamisega erinevates tööjärjekordades, on näiteks praimeridisain multiplex-PCR-i jaoks ja praimerite 5'-otsa pikendamine kasutaja valitud järjestusega, mis on vajalik mitmete biotehnoloogiliste meetodite jaoks (Rozen ja Skaletsky, 2000). Samuti ei teosta Primer3 ise sisendjärjestuses ebasoovitavate praimerite seondumiskohtade maskeerimist. Selleks on vajalikud programmid nagu GENOMEMASKER (Andreson *et al.*, 2006), RepeatMasker (http://www.repeatmasker.org) või teised.

1.2 K-meeride sagedustel põhinev järjestuse maskeerimine

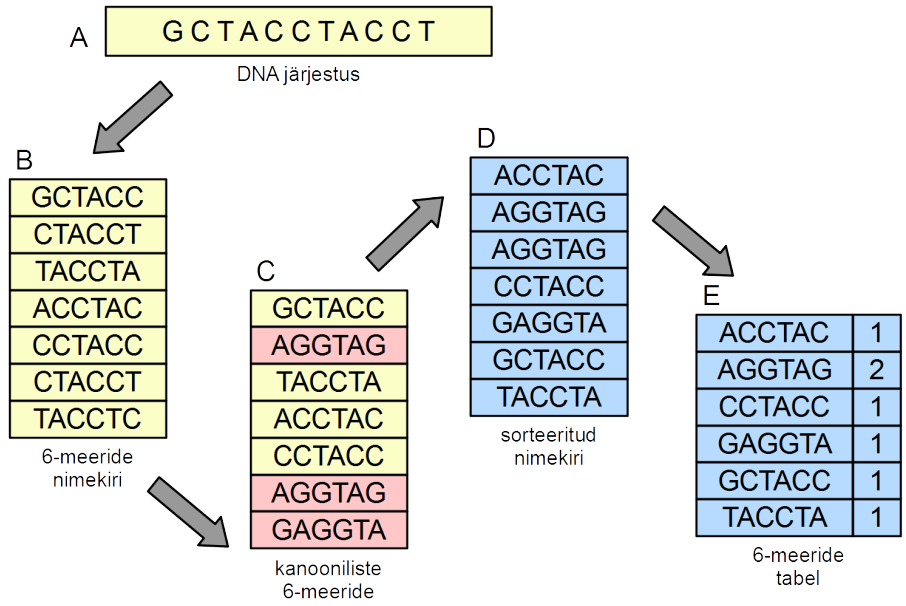
Praimerite mittespetsiifiliste seondumiste vältimiseks saab Primer3 programmile ette anda kordusjärjestusi sisaldava andmebaasi. Sellisel juhul Primer3-e algoritmi kohaselt nendele järjestustele praimereid ei disainita. Suure andmebaasi ning paljude praimerite korral on iga potentsiaalse praimeri võrdlemisel etteantud andmebaasiga tegemist aga küllaltki ajakuluka protsessiga. Samuti on kordusjärjestuste andmed tihti ebatäielikud või paljude organismide puhul hoopis puudu. Alternatiivseks variandiks on kasutada organismi k-meeride sagedusi.

Fikseeritud pikkusega k nukleiinhappejärjestuse fragmenti nimetatakse k-meeriks. K-meeride loendamine on kõigi mingis järjestuses (ka osaliselt ülekattes) olevate k-meeride arvude leidmine. Selleks on loodud mitmeid efektiivseid algoritme. Hetkeseisuga on kõige kiiremad k-meeride loendamiseks mõeldud programmid KMC2 (Deorowicz *et al.*, 2014), DSK2 (Rizk *et al.*, 2013), Jellyfish (Marçais ja Kingsford, 2011) ja GenomeTester4 (Kaplinski *et al.*, 2015).

GENOMEMASKER (Andreson *et al.*, 2006) on programmide pakett nukleiinhapete järjestustes korduvate alade maskeerimiseks, mis ei vaja eelnevalt teadaolevate kordusjärjestuste andmebaasi. Selle asemel kasutab nimetatud paketis olev programm *gmasker* lähte-DNA järjestuse maskeerimisel programmi *glistmakeri* loodud nn üleesindatud k-meeride nimekirja (ingl. k. *blacklist*). Enamasti on see nimekiri koostatud kogu uuritava genoomi järjestust kasutades. *gmasker* liigub libiseva aknaga üle DNA järjestuse ning maskeerib need k-meerid, mis leiduvad etteantud nimekirjas. Vaikimisi maskeeritakse ainult k-meeri 3'-otsa esimene nukleotiid. Kasutaja võib programmil lasta aga järjestust maskeerida kogu hetkel aknas oleva k-meeri ulatuses. Teise osana sisaldab GENOMEMASKER-i pakett ka programmi elektroonilise PCR-i läbiviimiseks.

GenomeTester4 (Kaplinski *et al.*, 2015) on algselt loodud GENOMEMASKER (Andreson *et al.*, 2006) paketis oleva programmi *glistmaker* uue implementatsioonina. Hetkeseisuga (03.11.2015) sisaldab GenomeTester4 kolme erinevat funktsionaalsust: GListMaker k-meeride loendamiseks ja sagedustabeli moodustamiseks, GListQuery varem loodud sagedustabelist k-meeride otsimiseks ning GListCompare k-meeride tabelitega hulgateoreetiliste tehete (ühend, ühisosa, vahe) tegemiseks. Käesoleva töö kontekstis on neist huvipakkuvateks eelkõige kaks esimest.

GListMaker leiab etteantud FastA või FastQ formaadis olevast järjestusest k-meeride sagedused ning salvestab need binaarsel kujul väljundfaili. K-meeride lugemiseks kasutab programm libisevat akent. Iga parasjagu aknas olev k-meer teisendatakse 64-bitiseks täisarvuks ning salvestatakse ajutisse k-meeride nimekirja, kusjuures igale k-meerile ja tema pööratud komplementaarsele k-meerile vastab sama täisarv. Seejärel sorteeritakse saadud nimekiri, kasutades lineaarse keerukusega sorteerimisalgortmi *in-place radix sort* (Duvanenko, 2009). Viimase sammuna arvutatakse kõigi k-meeride sagedused, käies järjestatud nimekiri veel kord algusest lõpuni läbi, ning salvestatakse saadud tulemused binaarsesse faili (vt joonis 3). Algne GENOMEMASKER-i paketti kuuluv programm salvestas vaid need k-meerid, mille sagedus oli suurem kasutaja etteantud piirmäärast. Uuem programm salvestab kõigi huvipakkuvas järjestuses leiduvate k-meeride puhul nii k-meeri enda kui ka tema sageduse. Seega saab juba loodud k-meeride tabeleid kasutada korduvalt väga erinevate ülesannete lahendamiseks. Olulised erinevused on veel näiteks need, et GListMaker töötab mitmel lõimel, lubab sisendina ka FastQ formaadis faile ning



***Joonis 3: GListMakeri tööpõhimõte** 6**-meeride näitel:** DNA järjestusest (A) loetakse järjest kõik k-meerid (B). Vajadusel teisendatakse k-meere nii, et salvestatud kirje oleks kahest antikomplementaarsest järjestusest leksikograafiliselt väiksem (nn kanooniline) (C). Seejärel k-meerid järjestatakse (D) ning leitakse iga k-meeri sagedus (E). Saadud tabel kirjutatakse binaarsel kujul faili. (Lepamets, 2014)*

töötab failidega, mis on suuremad kui 4GB. Samuti, kui vanem programm loendas vaid kuni 16-nukleotiidi pikkuseid k-meere, siis GListMakeri puhul on maksimaalseks lubatud k-meeri pikkuseks 32.

GListQuery on mõeldud GListMaker-iga loodud k-meeride tabelitest otsingute teostamiseks. Programm kasutab logaritmilise keerukusega kahendotsingu algoritmi ning tagastab otsitava k-meeri sageduse tabelist. Kasutaja saab huvipakkuvaid k-meere programmile anda käsureaparameetriga, failina üksteise alla kirjutatult või teise k-meeride tabelina. Ühtlasi võimaldab GListQuery võtta sisendiks FastA või FastQ faili ning otsida tabelist kõiki etteantud järjestuses leiduvaid k-meere. GListQuery-t saab kasutada ka kõigi selliste k-meeride sageduste leidmiseks, mis erinevad etteantud k-meerist kuni mingi positsioonide arvu võrra. Sellisel juhul genereeritakse k-meerist kõik nii mitme asendusega variandid, otsitakse tabelist kõiki saadud variante ning tagastatakse sageduste summa.

2 Töö eesmärgid

Primer3 on üks laialdasemalt kasutatavaid praimeridisaini programme. Selle algoritmi on erinevate töörühmade poolt järk-järgult täiendatud ning hetkeseisuga arvestab see peaaegu kõigi kindlakstehtud praimeridisaini kriteeriumitega. Küll aga ei arvesta Primer3 praimerikandidaatide potentsiaalsete seondumiskohtade arvu uuritavas genoomis. PCR-i metoodikast tuleneb, et praimerid, millel on palju mittespetsiifilisi seondumisi uuritava genoomi piires, annavad suurema tõenäosusega valeprodukti või ei tööta üldse (Ghedira *et al.*, 2009). Seetõttu on oluline antud informatsiooni kaasamine praimeridisaini loogikasse.

Seni on tavapäraseks tööjärjekorraks olnud esmalt lähtejärjestuses kordustega ala-de maskeerimine (näiteks programmiga RepeatMasker) ja seejärel maskeeritud järjestuse-le praimerite disainimine. Korduste maskeerimine ei taga aga alati disainitud praimeri spetsiifilise seondumise lähte-DNA-le. Seetõttu kuulub tööjärjekorda ka mõne eraldiseis-va joondamisprogrammiga (näiteks BLAST, Altschul *et al.* (1997)) uuritavast genoomist mittespetsiifiliste seondumiskohtade leidmine (näiteks Kuhn *et al.* (2015)). See aga on tü-likas ja tähendab, et ebasobivate praimerite asemele tuleb vajadusel disainida uued ning neid ka uuesti kontrollida. Seondumiskohtade arvu kaasamine praimeridisaini kiirendaks kirjeldatud protsessi tunduvalt.

Praimerite seondumiskohtade hindamiseks ei piisa disainialgoritmil ainult sisend-järjestusest, vaid vaja on informatsiooni kogu uuritava genoomi kohta. Mõistlik ja lihtne on kasutada uuritava genoomi k-meeride sagedustabeleid, mis juba iseenesest sisaldavad informatsiooni lühikeste genoomipiirkondade korduste arvude kohta.

Käesoleva töö eesmärgid on püstitatud järgnevalt:

1. Koostada statistiline mudel, mis kasutaks k-meeride tabeleid ja hindaks praimeri mittetöötamise tõenäosust, lähtudes selle potentsiaalsete seondumiskohtade arvust uuritavas genoomis.

2. Uuesti implementeerida GENOMEMASKER-i paketti kuuluv *gmasker*-i programm selliselt, et see kasutaks GenomeTester4 loodud k-meeride tabeleid ja eelmises punk-tis kirjeldatud praimerite mittetöötamise tõenäosuse mudelit ning maskeeriks si-sendjärjestusest piirkonnad, millega ülekattes olevate praimerite mittetöötamise tõenäosused on kasutaja määratud piirnivoost suuremad.

3. Integreerida loodud maskeerimisalgoritm Primer3-e disainiloogikasse.

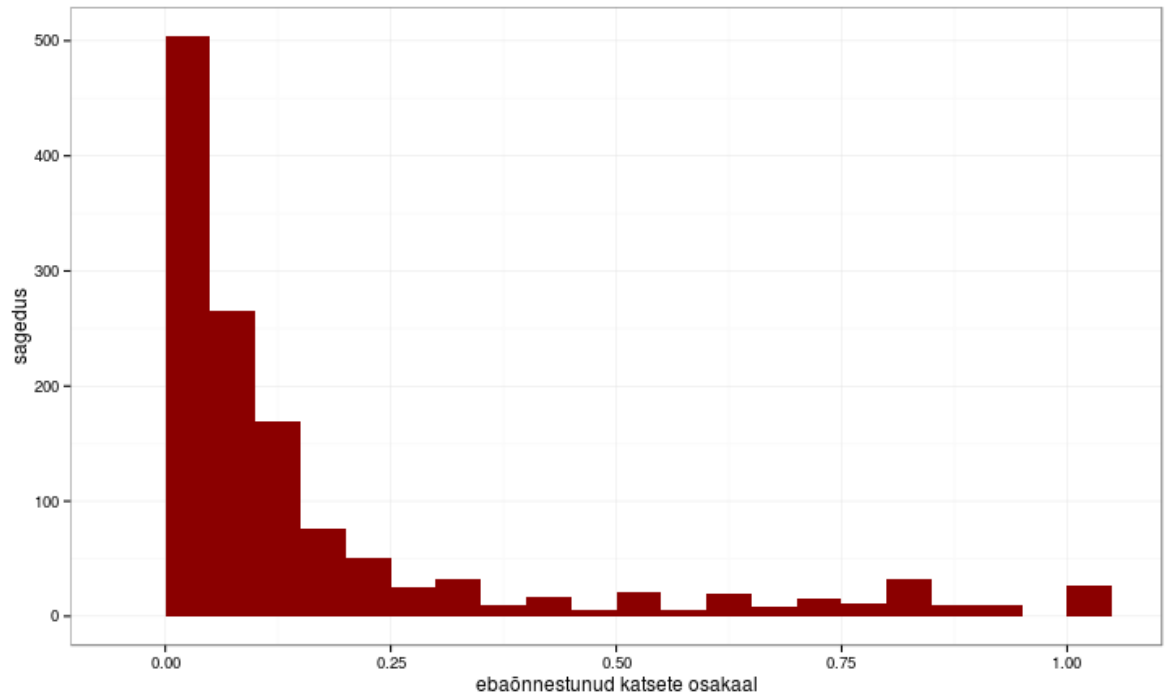
4. Testida loodud maskeerimisalgoritmi efektiivsust ja võrrelda disainitud praimereid juhtudel, kus maskeerimist kasutati ja kus ei kasutatud.

3 Metoodika ja implementatsioon

3.1 Praimerite andmestik

Käesolevas töös on kasutatud inimese genoomile disainitud 1313 PCR-i praimeripaari. Nende hulgas on 300 praimeripaari, mis on disainitud juhuslikele piirkondadele genoomis. Ülejäänud praimerid on disainitud veidi enam kui tuhandele 22. kromosoomis leiduvale ühenukleotiidsele variatsioonile (Dawson *et al.*, 2002). Kõik andmestikus olevad praimerid on järjestuselt erinevad ning ükski praimer pole teisega osalises ega täielikus ülekattes. Praimerite pikkused varieeruvad vahemikus 18 kuni 25 nukleotiidi, sulamistemperatuurid jäävad vahemikku 48°C kuni 75°C ja GC-protsendid on 27% kuni 80%. Praimerite disainil on arvestatud kõigi teadaolevate valikukriteeriumitega. Välja on jäetud vaid kontrollimine, kas praimerid on mõne teadaoleva kordusjärjestusega ülekattes. Seega sobib käesolev andmestik hästi hindamaks seondumiskohtade arvu mõju praimerite töökindlusele.

Kõiki nimetatud praimeripaaride töötamist testiti laboris. Selleks viidi iga paariga läbi 10 kuni 185 PCR-i reaktsiooni ning märgiti üles õnnestunud ja ebaõnnestunud katsete arvud. Täpne laboriprotokoll on toodud Andreson *et al.* (2008). Iga praimeripaari korral võeti laborikatsete pealt hinnatud praimeri mittetöötamise tõenäosus võrdseks antud praimeritega ebaõnnestunud reaktsioonide osakaaluga (vt ka joonis 4). Edaspidises töös



Joonis 4: PCR-i katsetulemuste andmestikus olevate praimeripaaride ebaõnnestunud reaktsioonide osakaalude histogramm

käsitletakse seda kui praimerite tegelikku mittetöötamise tõenäosust.

Andmestikus olevate praimerite potentsiaalsete seondumiskohtade arvu inimese genoomis hinnati nende 3'-otsa k-meeride põhjal. Näiteks praimerite CAGTGGAAGCACTCTTTTGG ja CAGAATTGTCGTGGATTGGG ning $k = 10$ korral on vastavad 3'-otsa k-meerid ACTCTTTTGG ja GTGGATTGGG. Iga fikseeritud k korral iseloomustab praimeripaari kaks k-meeri, üks kummagi praimeri jaoks. Igale andmestikus olevale praimerile leiti vastavate k-meeride sagedused inimese referentsgenoomis (versioon GRCh38) kõigi täisarvude $k = [6, 16]$ korral. Et nimetatud sageduste väärtuste vahemikud on väga laiad, siis teisendati need logaritmilisele skaalale. Seejärel defineeriti muutujad K6...K16, mille väärtused on iga praimeripaari puhul kahest logaritmitud k-meeri sagedusest suuremad. Jätkates eelmist näidet, kui leitakse, et ACTCTTTTGG paikneb inimese genoomis 12 korda ja GTGGATTGGG 34 korda, siis K10 saab väärtuseks log(34). Lisaks leiti analoogiliselt muutujate K6_1MM...K16_1MM ja K6_2MM...K16_2MM väärtused, milleks on selliste k-meeride logaritmitud summaarsed sagedused inimese referentsgenoomis, mis erinevad vaadeldava praimeri 3'-otsa k-meerist vastavalt kuni ühe ja kuni kahe nukleotiidi võrra. Kõigi k-meeride sageduste leidmiseks kasutati GenomeTester4 paketi programme GListMaker ja GListQuery (Kaplinski *et al.*, 2015). Näide PCR-i katsetulemuste andmestikust on toodud lisas A.

3.2 Praimeri mittetöötamist hindava mudeli koostamine

3.2.1 Praimeri mittetöötamise kirjeldamine logistilise regressiooni abil

Käesoleva töö käigus koostati mudel PCR praimerite mittetöötamise hindamiseks, lähtudes praimerite potentsiaalsete seondumiskohtade arvust.

Olgu y binaarne tunnus, kus

$$y = \begin{cases} 0, & \text{kui praimer töötab} \\ 1, & \text{kui praimer ei tööta} \end{cases}$$

Sellisel juhul on y Bernoulli jaotusega parameetriga p, mis kirjeldab praimeri mittetöötamise (1 saamise) tõenäosust. Praimerite seondumiskohtade arvu kirjeldavad eelmises alapeatükis mainitud tunnused K6...K16, K6_1MM...K16_1MM, K6_2MM...K16_2MM. Osades mudelites kasutati ka ruutliikmeid, näiteks K6∗K6. Tähistagu X kõigi nimetatud k-meeride sageduste informatsiooni sisaldavate tunnuste hulka.

Praimerite mittetöötamise tõenäosuse hindamiseks kasutati üldistatud lineaarset mudelit logit seosefunktsiooniga. Hariliku lineaarse regressiooni korral hinnataks sõltuvat tunnust y mudeliga

$$y = a + b_1 x_1 + \ldots + b_n x_n,$$

kus $(x_1, \ldots, x_n)$ on mingi alamhulk tunnuste hulgast X, a on mudeli vabaliige ning $(b_1, \ldots, b_n)$ on tunnustele vastavad koefitsiendid. Sellise mudeli korral võib tunnus y omandada väärtuseid tervest reaalarvude hulgast. Üldistatud lineaarse mudeli korral rakendatakse tunnusele y mingit seosefunktsiooni, mis muuhulgas võimaldab uuritava tunnuse väärtuseid piirata. Näiteks kui soovitakse hinnata tõenäosust, peab see jääma vahemikku $[0, 1]$. Töös kasutusel oleva logit seosefunktsiooni korral

$$a + b_1 x_1 + \ldots + b_n x_n = \text{logit}(\mu) = \log(\frac{\mu}{1-\mu}),$$

kus $\mu = E(y)$ on y keskväärtus, mis on antud juhul võrdne vaadeldava praimeri mittetöötamise tõenäosusega p, mis omakorda avaldub ülaltoodud seosest

$$p = \frac{e^{a+b_1 x_1 + \ldots + b_n x_n}}{1 + e^{a+b_1 x_1 + \ldots + b_n x_n}}.$$

Kirjeldatud meetodit nimetatakse ka logistiliseks regressiooniks.

Katsetulemuste valimis on ühte ja sama praimeripaari testitud korduvalt ning seega pole kõik vaatlused omavahel sõltumatud. See aga rikub üldistatud lineaarsete mudelite sõltumatuse eeldust. Seetõttu kasutati käesolevas töös mudeli valimisel ja koefitsientide hindamisel üldistatud lineaarse mudeli asemel üldistatud lineaarset segamudelit

$$p = \frac{e^{a+b_1 x_1 + \ldots + b_n x_n + C}}{1 + e^{a+b_1 x_1 + \ldots + b_n x_n + C}},$$

kus C kirjeldab uuritava praimeripaari juhuslikku efekti. Üldistatud lineaarse segamudeli sobitamiseks kasutati statistikaprogramm R-i paketti glmmADMB.

3.2.2 Tunnuste valimine mudelisse

Olgu X defineeritud kui kõigi selliste tunnuste hulk, mida mudeli valiku algoritm potentsiaalsete kirjeldavate muutujatena arvesse võtab. See tähendab, et algoritmi väljastatud tunnused, mida mudelis kasutatakse, moodustavad X mingi alamhulga. Mudeli valimiseks kasutati modifitseeritud varianti meetodist, mis näeb ette mudelisse ükshaaval uute tunnuste lisamist seni, kuni selline tegevus mudelit parandab (ingl. k. *stepwise forward selection*). Mudeli headuse hindamiseks kasutati Akaike informatsioonikriteeriumi (AIC). AIC hindab mudeli kvaliteeti ning avaldub kujul

$$AIC = -2\log(L) + 2k_t,$$

kus L on mudeli tõepärafunktsiooni maksimaalne väärtus ja k_t on tunnuste arv mudelis. Võttes arvesse tunnuste arvu, aitab AIC vältida mudeli ülesobitamist. Eelkõige kasutatakse AIC väärtust selleks, et mitme mudeli hulgast parim valida. Eelistada tuleks seda mudelit, mille AIC väärtus on vähim. Kui kahe mudeli AIC väärtuste vahe on väiksem

kui 2, siis pole antud treeningandmestikku kasutades võimalik öelda, kumb mudelitest on parem (Burnham ja Anderson, 2002).

Enne mudeli valiku algoritmi rakendamist eemaldati praimerite andmestikust juhuslikult 400 vaatlust, mida kasutati hiljem mudeli valideerimiseks. Ülejäänud 913 vaatlust moodustas treeningandmestiku. Mudeli valiku algoritm lähtus järgmistest tingimustest:

- Mudel ei sisalda rohkem kui kahe erineva pikkusega k-meeride sagedusi.
- Mudelis olevate tunnuste komplekt minimiseerib mudeli AIC väärtust.

Algoritm võtab sisendiks tunnuste hulga X, treeningandmestiku ning täisarvu m, mis tähistab, mitme erineva pikkusega k-meeride sagedusi mudel maksimaalselt võib sisaldada. Mingi tunnusega f_1 sarnane tunnus f_2 on defineeritud kui tunnus, mille väärtus on leitav f_1-ga sama pikkusega k-meeride sagedusi arvestades ja $f_1 \neq f_2$. Näiteks tunnused K6, K6_1MM ja K6∗K6 on kõik omavahel sarnased, sest kõigi väärtuste leidmiseks kasutatakse 6-meeride sagedusi. Tähistagu $S(f_1)$ kõigi f_1-ga sarnaste tunnuste hulka ning olgu $S(F) = \bigcup_{f \in F} S(f)$, $F \subset X$.

Olgu U juba mudelisse valitud tunnuste hulk. Defineerime hulgad $W = S(U)$ ja $V = X \setminus (U \cup W)$. Tähistagu $M(U)$ sellist üldistatud lineaarset segamudelit, mis sisaldab kõiki kirjeldavaid muutujaid hulgast U ja $AIC(M(U))$ selle mudeli AIC väärtust. Mudeli valimise algoritmi kirjeldus on järgmine:

1. $U = \emptyset$, $V = X$, $W = \emptyset$
2. Leitakse
$$f_0 = \arg\min_{f \in V_0}(AIC(M(U \cup \{f\})) - AIC(M(U))),$$
kus $V_0 = \{f \in V : AIC(M(U \cup \{f\})) < AIC(M(U))\}$. Kui $V_0 = \emptyset$, siis algoritm lõpetab töö.
3. $U = U \cup \{f_0\}$, $W = W \cup S(f_0)$, $V = V \setminus (\{f_0\} \cup S(f_0))$
4. Leitakse
$$f_0 = \arg\min_{f \in W_0}(AIC(M(U \cup \{f\})) - AIC(M(U))),$$
kus $W_0 = \{f \in W : AIC(M(U \cup \{f\})) < AIC(M(U))\}$.
5. $U = U \cup \{f_0\}$, $W = W \setminus \{f_0\}$
6. Algoritmisamme 4. - 5. korratakse, kuni $W_0 \neq \emptyset$.

7. Algoritmisamme 2. - 6. korratakse, kuni mudelis olevad tunnused ei kasuta rohkem kui m erineva pikkusega k-meeride sagedusi.

Kirjeldatud algoritm on oma tüübilt ahne. See tähendab, et igal tsüklisammul lisatakse mudelisse see tunnus, mis sel hetkel mudeli AIC väärtust minimiseerib. Algoritm väljastab ühe mudeli. Tihti ei pruugi aga taoline lähenemine anda teoreetiliselt parimat mudelit. Et algoritmi valikukriteeriumit veidi lõdvendada ning saada väljundiks kõik enam-vähem võrdse headusega mudelid, muudeti algoritm rekursiivseks hargnemistega 3. ja 5. sammu juures. Muudetud algoritmi versioonis leitakse enne uue tunnuse f_0 mudelisse lisamist kõik sellised tunnused $f \in V$ (5. sammu puhul $f \in W$), mille korral kehtivad

$$AIC(M(\{U \cup f\})) < AIC(M(U))$$

ja

$$AIC(M(\{U \cup f\})) < AIC(M(\{U \cup f_0\})) + 2.$$

Seejärel hargneb algoritm nii mitmeks haruks, kui palju on sobivaid tunnuseid f ning igas algoritmi harus lisatakse mudelisse neist tunnustest üks. Seejärel jätkub algoritm tavapäraselt kuni järgmise hargnemiseni (algoritmi kood on kättesaadav GitHubi repositooriumist https://github.com/maarjalepamets/masker).

Kirjeldatud algoritmiga leiti AIC järgi parimaid mudeleid, alustades neljast erinevast tunnuste komplektist X:

I. X sisaldab kõiki praimerite andmestikus olevaid k-meeride sagedusi arvestavaid tunnuseid ja nende ruute.

II. X sisaldab tunnuseid K6...K16, K6_1MM...K16_1MM ja nende ruute.

III. X sisaldab tunnuseid K6...K16 ja nende ruute.

IV. X sisaldab tunnuseid K6...K16, kuid mitte nende ruute.

Saadud mudelite hulgast valiti väike alamhulk, mida võrreldi omavahel ja Andreson *et al.* (2008) mudelitega (millest oli eemaldatud GC-sisaldust kirjeldav tunnus). Valiku tegemisel jälgiti, et valitud saaks vähemalt üks mudel igast grupist. Väga sarnaste mudelite korral valiti võrdlusesse selliseid, mille AIC väärtus on väiksem või mis kasutavad lühemaid k-meeride pikkuseid.

Valikus olevate mudelite võrdlemiseks kasutati analüüsi alguses valimist eemaldatud 400 vaatlust. Lõplik mudel valiti selle alusel, kuivõrd ühe või teise mudeli antud hinnangud praimerite mittetöötamise tõenäosusele aitasid testandmestikust halvasti töötavaid praimereid välja selekteerida. Lisaks arvestati seda, kui kiiresti on erinevate mudelitega võimalik arvutusi teha ja neis leiduvate tunnuste väärtuseid leida, sest eesmärgiks

on mudeli integreerimine praimeridisaini konveierisse, mille töö ei tohiks mudeli kasutamise tõttu oluliselt aeglustuda. Arvutuste aegu mõõdeti Tartu Ülikooli bioinformaatika õppetooli CentOS 5.10 Linuxi serveris, millel on 32 tuuma taktsagedusega 2,27 GHz ja 512 GB operatiivmälu.

3.2.3 Bioloogiliselt relevantse vabaliikme leidmine

Oluline on arvestada, et antud algoritm maksimiseerib loodud regressioonmudeli suurima tõepära hinnangut valimis esindatud väärtuste vahemikus. Väljaspool andmestikus esindatud väärtusi võib mudeli ennustus oodatavast oluliselt erineda. Kasutades bioloogilist taustateadmist PCR reaktsiooni metoodikast, on võimalik kontrollida, kas mudeli ennustused on ekstreemsetel juhtudel relevantsed.

Antud töös osutusid probleemseteks ruutliikmeid sisaldavad mudelid, milles kasutatakse k-meere, mille pikkus on väiksem kui 14 nukleotiidi. Näiteks katavad praimerite andmestikus tunnuse K11 väärtused lõigu [5,22; 13,52]. Teoreetiliselt aga võivad potentsiaalsed K11 väärtused muutuda vahemikus $[0, \infty)$. Väga suurtele k-meeride sagedustele vastavad väga kõrged mittetöötamise tõenäosused ning nende täpne väärtus kasutajat enamasti ei huvita, kuni mudel ennustab nad kõrgeteks. Küll aga tekib probleem, kui mudel hindab praimeri, mille 3'-otsa k-meeri sagedus genoomis on väga madal, peaaegu või täielikult kasutuskõlbmatuks.

Selle probleemi vältimiseks hinnati ruutliikmeid sisaldava mudeli puhul mudeli vabaliige ülejäänud mudeli koefitsientidest eraldi. Arvutuskäik on toodud sellise mudeli näitel, mis sisaldab tunnust K11. Esmalt leiti 10% kõige madalamate K11 tunnuste väärtustega praimeripaaride tegelike mittetöötamise tõenäosuste mediaan m (K11 korral $m \approx 0{,}07$). Pole teada, milline on sellise praimeri mittetöötamise tõenäosus, mille 3'-otsas olev 11-meer on uuritavas genoomis unikaalne, kuid PCR reaktsiooni mehhanismi aluseks võttes ei peaks see olema suurem kui m. Sellest eeldusest lähtudes leiti mudelile vabaliige järgmiselt:

$$a = \text{logit}(m) = \log(\frac{m}{1-m}).$$

K11 puhul $a \approx -2{,}59$. Ülejäänud koefitsiendid hinnati juba etteantud vabaliikme väärtusest lähtudes.

3.3 Maskeerimisalgoritm

Käesoleva töö raames implementeeriti algoritm, mis maskeerib etteantud lähtejärjestuses need piirkonnad, millega ülekattes olevate praimerite hinnanguline seondumiskohtade arv uuritava organismi genoomis on sedavõrd suur, et pärsib PCR-i reaktsiooni edukat toimumist. Selleks kasutab algoritm selle organismi genoomses järjestuses leiduvate k-meeride

sagedusi ning mudelit, mis hindab nendest sagedustest lähtuvalt praimeri töökindlust. Vaikimisi kasutatakse mudelit, mille kuju on leitud, kasutades eelnevalt kirjeldatud praimerite andmestikku ja metoodikat.

Maskeerimise funktsionaalsus on implementeeritud funktsioonide kogumikuna, mida on lihtne integreerida Primer3 programmi koodi. Lisaks on algoritmi võimalik kompileerida ka eraldi käivitatava programmina. Viimane annab kasutajale võimaluse varieerida mitmete parameetrite väärtuseid (vt peatükk 3.3.1), mis Primer3-e versioonis on fikseeritud, ning kasutada ka enda loodud k-meeride sagedustel põhinevaid mudeleid praimerite mittetöötamise tõenäosuste hindamiseks. Lisaks võimaldab eraldi käivitatav programm mudeli kasutamisest üldse loobuda ning maskeerida järjestuses need k-meerid, mida leidub etteantud k-meeride tabelis kasutaja määratud sagedusega võrdselt või rohkem.

Loodud maskeerimisalgoritm baseerub GenomeTester4 paketi (Kaplinski *et al.*, 2015) k-meeride tabelitel. Vajalikud tabelid tuleb eelnevalt eraldi luua GListMaker programmiga. Programm on implementeeritud programmeerimiskeeles C (Kernighan ja Ritchie, 1988) ning on tulevikus kättesaadav Primer3-e SourceForge'i repositooriumist (aadressil http://primer3.sourceforge.net/, Primer3-e täiustatud versiooni ja maskeerimise programmi ajutine asukoht on https://github.com/maarjalepamets/masker). Programm (ka eraldi käivitatav versioon) ei kompileeru Primer3-st sõltumatult, sest kasutab veateadete haldamiseks viimase funktsioone ja loogikat. Loodud rakendus on eraldiseisvana käivitatav käsurealt. Programmi kompileerumist ja kasutamist on testitud operatsioonisüsteemides Linux CentOS 5.10, Linux Ubuntu 13.10 ja Mac OS X.

3.3.1 Sisend ja väljund

Käsurealt käivitatav maskeerimise programm võtab sisendina maskeeritava järjestuse, praimerite mittetöötamise tõenäosust hindava mudeli ning arvulised parameetrid, mis määravad maskeerimise viisi, ranguse ja ulatuse. Parameetrite täisnimekiri ja käsurea näited on toodud töö lisades (vt lisasid C ja D). Programmi väljundiks on sisendjärjestusega samas formaadis maskeeritud järjestus, mis trükitakse standardväljundisse.

Sisendjärjestuse saab programmile anda FastA formaadis või lihtsa järjestuse failina. Vastava faili puudumisel jääb programm ootama standardsisendit klaviatuurilt. Mudeli määramiseks on kolm erinevat varianti.

Mudeli tunnused võib määrata ükshaaval käsurealt. Sel juhul tuleb iga tunnuse jaoks täpsustada k-meeride tabel, mis sisaldaks huvipakkuva pikkusega k-meeride sagedusi; kas tunnuse väärtuse arvutamisel vaadatakse täpseid või kuni ühe/kahe nukleotiidi võrra erinevate k-meeride sagedusi; kas tegemist on mudelis lineaar- või ruutliikmega ning mis on antud tunnusele vastav mudeli koefitsient. Vaikimisi on koefitsiendi väärtuseks 1,0. Kui muutuja on määratud lihtsalt kui üks reaalarv, siis programm eeldab, et tegemist on

mudeli vabaliikmega (vt lisa D näide 1). Eraldi on võimalik järjestust maskeerida, kasutades ainult ühte k-meeride tabelit ja seal leiduvate k-meeride sageduste absoluutarve. Sel juhul ignoreeritakse kõiki tunnust määravaid käsurea elemente peale k-meeride tabeli nime (vt lisa D näide 6).

Muutujate ükshaaval määramise asemel võib kõik muutujate deklaratsioonid koondada ühte faili ning see programmile sisendiks anda (vt lisa D näide 2). See võimaldab sama mudelit lihtsasti kasutada mitme erineva maskeerimisülesande jaoks. Samuti võib muutujaid deklareerida segamini failist ja käsurealt. Sel juhul saab sama tunnuse mitmekordse esitamise korral kõrgema prioriteedi käsurealt määratud mudeli koefitsient.

Kolmas variant on kasutada maskeerimisalgoritmi vaikimisi määratud mudelit. Sellisel juhul on kasutatavate k-meeride pikkused, neile vastavate tunnuste kordajad ja muu vajalik informatsioon programmile juba teada. Kasutaja peab määrama vaid nende failide eesliite, mis sisaldavad vajaminevate k-meeride sagedusi (vt lisa D näide 3). Mõistlik on k-meeride tabelite eesliitena kasutada näiteks selle organismi nime või muud identifikaatorit, kelle genoomist antud tabel on loodud.

Lisaks järjestusele ja mudelile saab programmile käsurealt määrata praimeri mittetöötamise tõenäosuse nivoo, millest alates nukleotiide maskeeritakse (vaikimisi väärtus on 0,2). Vaikimisi maskeeritakse vaid vaadeldava k-meeri 3'-otsmine nukleotiid, kuid kasutaja saab soovi korral täpsustada, mitu nukleotiidi ta k-meeri 3'-otsast kummaski suunas maskeerida tahab (vt lisa D näide 4). Programm võib lähtejärjestust maskeerida kahel viisil: asendades maskeeritud nukleotiidid mingi fikseeritud tähemärgiga (*hard-masking*) või sama nukleotiidi tähistava väiketähega (*soft-masking*). Ka seda saab kasutaja käsurealt määrata (vt lisa D näiteid 4 ja 5). Vaikimisi maskeeritakse nukleotiide tähemärgiga 'N'. Ühtlasi on see kasutaja otsustada, kas maskeeritakse etteantud nukleotiidse piirkonna ühe, teise või mõlema ahela järgi (vt lisa D näide 5).

Primer3-e maskeerimisalgoritmi versioon erineb käsurealt käivitatavast versioonist selle poolest, et nii sisend- kui väljundjärjestuse näol on tegemist sõne tüüpi muutujatega (C dünaamiliselt allokeeritud tähemärkide massiiv), mida Primer3-e töövoos eri funktsioonide vahel liigutatakse. Maskeerimiseks kasutatakse selles versioonis alati vaikimisi mudelit ning maskeeritakse väiketähega. Kasutaja saab määrata maskeerimisnivoo, maskeeritavate nukleotiidide arvu ja vajaminevate k-meeri tabelite eesliite.

3.3.2 Ringpuhver

Algoritmi keskseks struktuuriks on ringpuhver, millesse järjestusest loetavad tähemärgid ja muu järjestusepõhine informatsioon ajutiselt salvestatakse. See on vajalik kahel põhjusel. Esiteks võimaldab programm maskeerida suvalise arvu nukleotiide kummalegi poole uuritava k-meeri 3'-otsast. See tähendab, et sisendjärjestusest uue nukleotiidi lugemisel

pole algoritmil alati veel teada, kas antud nukleotiid tuleb maskeerida või mitte. Puhver võimaldab loetud nukleotiide talletada seni, kuni vajalik informatsioon on laekunud. Lisaks võib sisendjärjestus sisaldada peale nukleotiidide ka teisi tähemärke. Neist mõndade (näiteks reavahetused ja tühikud) puhul loetakse k-meere üle antud tähemärgi (k-meeri järjestus võib alata ühel real ja lõppeda järgmisel). Puhver võimaldab programmil maskeeritud järjestuse väljutada sisendjärjestusega täpselt samas formaadis.

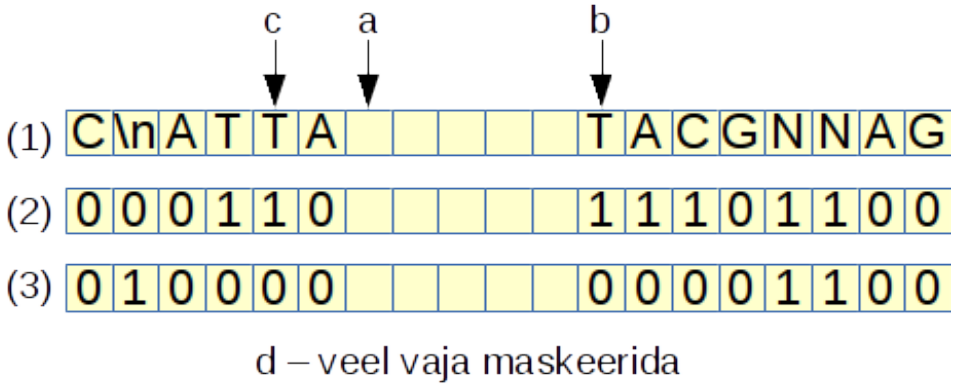
Puhver koosneb neljast võrdse suurusega massiivist ja neljast täisarvust. Esimene massiiv sisaldab sisendjärjestuse seda osa, mida algoritm on juba analüüsinud, kuid pole veel väljundisse saatnud. Ülejäänud massiivides on elementideks binaarsed väärtused 0 või 1. Kaks neist (üks massiiv kummagi ahela jaoks) sisaldavad informatsiooni selle kohta, millised järjestuse massiivis olevad nukleotiidid tuleb väljastada maskeeritult (väärtus on 1). Neljandas massiivis on märgitud kõik järjestuse massiivi need positsioonid, milles olev tähemärk ei vasta ühelegi nukleotiidile (väärtus on 1). Puhvri struktuuri kuuluvatest täisarvudest kolm osutavad massiivi nendele positsioonidele, kuhu lisatakse järgmine tähemärk, kust loetakse järgmine tähemärk ning kus asub selline viimati lisatud nukleotiid, mille maskeerimise staatus on juba kindlaks määratud. Neljandas täisarvus talletatakse nende nukleotiidide arv, mis tuleb veel mitteloetud järjestuse piirkonnast maskeerida (vt ka joonis 5.

Puhvri struktuuriga koos on implementeeritud hulk funktsioone puhvri täitmiseks, tühjendamiseks ja modifitseerimiseks. Iga järgmise tähemärgi puhvrisse lisamisel uuendatakse vajadusel puhvri kõiki massiive. Puhver tühjendatakse siis, kui algoritm üritab uut tähemärki sisestada puhvri sellisele positsioonile, milles olevat elementi pole veel väljundisse saadetud või kui ollakse jõudnud maskeeritava järjestuse lõppu. Tühjendamisel jäetakse puhvrisse alles vaid need nukleotiidid, mille maskeerimise staatus pole veel teada.

3.3.3 Järjestuse maskeerimine

Programmi käivitamisel kogutakse kasutaja määratud parameetrite väärtused selleks ettenähtud struktuuri ning luuakse ja algväärtustatakse ringpuhver. Praimerite mittetöötamise mudeli tunnuste loomisel kontrollitakse, kas kõik vajalikud k-meeride tabelid on olemas. Seejärel muudetakse nende tabelite kirjed programmile kättesaadavaks, kasutades *memory map* meetodit. Viimane tagab selle, et suuri tabeleid ei loeta tervenisti arvuti operatiivmällu, vaid piirdutakse nende osadega, mida programm parasjagu kasutab. Igale k-meeri tabelile lisatakse eeskiri, kuidas antud tabelit mudeli tunnuste väärtuste leidmiseks kasutada. Üks tabel võimaldab kuni kuue erineva tunnuse arvutamist. Näiteks, kasutades 16-meeride tabelit, saab leida väärtused tunnustele K16, K16_1MM, K16_2MM ja neid sisaldavatele ruutliikmetele (tunnuste tähistusi vt peatükk 3.1).

Olgu k_1 pikima ja $k_2, \ldots, k_n$ ülejäänud mudelis kasutusel olevate k-meeride pikku-



Joonis 5: Ringpuhver: *Puhver koosneb kokku neljast massiivist. Nendeks on järjestuse massiiv (1), kaks maskeerimisstaatuste massiivi (joonisel ainult üks (2)) ja massiivi, milles on tähistatud kõik need positsioonid, millel ei asu järjestuse massiivis nukleotiidi (3). Veel kuuluvad puhvri koosseisu viidad positsioonidele, kuhu järgmine nukleotiid lisatakse (a) ja kust järgmine nukleotiid väljastatakse (b). Lisaks jäetakse meelde, mitu nukleotiidi tuleb maskeerida veel puhvrisse sisestamata järjestuse osast (d). Et kasutaja võib soovida maskeerida laiemat piirkonda kui vaid üks nukleotiid, ei pruugi viimati lisatud nukleotiidide maskeerimisstaatused alati teada olla. Viit (c) osutab viimasele nukleotiidile, mille staatus on juba teada. Puhvri tühjendamisel väljastatakse positsioonid viitade (b) ja (c) vahel (otspunktid k.a.). Järjestuse positsioonid, mis ei vasta ühelegi nukleotiidile ega tühikule/reavahele (N), märgitakse koheselt maskeerituks.*

sed. Ühtlasi olgu α kasutaja määratud praimerite mittetöötamise tõenäosuse piirmäär, m_u praimeri 3'-otsast ülesvoolu (5'-suunas) maskeeritavate nukleotiidide arv ja m_d praimeri 3'-otsast allavoolu (3'-suunas) maskeeritavate nukleotiidide arv. Tähistagu M ringpuhvri seda massiivi, mis sisaldab maskeerimisinformatsiooni sisendjärjestuse 5'→3'-suunalise ahela jaoks, ja $M[i]$ selle massiivi i-ndat elementi.

Algoritmi töö käigus liigutakse sammhaaval üle etteantud sisendjärjestuse. Iga sammuga lisatakse puhvrisse juurde üks tähemärk, vajadusel puhvrit eelnevalt tühjendades. Olgu h viimati lisatud tähemärgi positsioon puhvris. Iga kord, kui algoritm on sisendjärjestusest lugenud järjest k_1 nukleotiidi (tühikuid ja reavahesid arvestamata) täidetakse järjestuse 5'→3'-suunalise ahela maskeerimiseks järgnev algoritm:

1. Leitakse loetud k_1-meerile vastavad $k_2 \ldots k_n$-meerid selliselt, et k_i-meeri ($i = 2 \ldots n$) järjestus vastab k_1-meeri k_i-le 3'-otsa nukleotiidile.

2. Iga k_j-meeri ($j = 1..n$) korral tehakse järgmised algoritmisammud:

   (a) Otsitakse kahendotsingu algoritmi abil vastava pikkusega k-meeride tabelist tunnuste väärtuste arvutamiseks vajalikud k-meeride sagedused. Näiteks $k_j =$

11 korral on tunnuse K11_1MM arvutamiseks vaja leida uuritava 11-meeri enda sageduse ja kõigi selliste 11-meeride, mis erinevad uuritavast 11-meerist ühe nukleotiidi võrra, sageduste summa.

(b) Arvutatakse mudeli tunnuste väärtused, logaritmides eelmises punktis leitud summasid.

3. Leitakse logaritmitud k-meeride sageduste lineaarkombinatsioon, kasutades tunnustele vastavaid koefitsiente. Saadud väärtusele liidetakse mudeli vabaliige.

4. Hinnatakse sellise praimeri mittetöötamise tõenäosust p, mille 3'-otsa järjestus vastab vaadeldavate k_j-meeride järjestustele, kasutades logistilise regressiooni mudelis eelnevalt leitud lineaarkombinatsiooni väärtust.

5. Kui $p > alpha$, siis $M[m_i] = 1$, iga sellise massiivi indeksi m_i puhul, mille korral kehtib $h \geq m_i > h - m_u$. Samuti jäetakse meelde, et veel mitteloetud järjestuse piirkonnast tuleb maskeerida m_d järgmist nukleotiidi.

Sisendjärjestuse 3'→5'-suunalise ahela maskeerimine toimub analoogiliselt. Erineb vaid k_j-meeride komplekt (ainult k_1-meeri järjestus on sama) ja maskeerimisinformatsiooni sisaldava massiivi uuendamine. Mõlema ahela maskeerimisel korratakse ülaltoodud algoritmisamme kahe erineva k_j-meeride komplektiga.

Käsurealt käivitatav programm võimaldab maskeerida ka ilma mudelita, kasutades ühe fikseeritud pikkusega k-meeride sagedusi. Sellisel juhul määrab kasutaja k-meeride sageduse piirmäära. Algoritm liigub libiseva aknaga üle maskeeritava järjestuse, kontrollib k-meeride tabelist kõigi järjestuses olevate k-meeride sagedusi ja maskeerib kõik sellised, mille leitud sagedus on piirmäärast suurem. Sarnaselt mudeliga maskeerimisele saab kasutaja valida maskeeritavate nukleotiidide arvu ja maskeerimise piirmäära.

3.4 Maskeerimisfunktsionaalsuse integreerimine Primer3 programmi

Primer3-le lisati kuus globaalset parameetrit (vt tabel 1). Üks neist on maskeerimisfunktsiooni sisse- ja väljalülitamiseks. Neli määravad maskeerimise ranguse (praimeri mittetöötamise tõenäosuse piirmäära), maskeeritavate nukleotiidide arvu ja selle, milliseid k-meeride tabeleid maskeerimiseks kasutatakse. Kuuenda parameetriga saab määrata, kui suure kaaluga praimeri mittetöötamise tõenäosust lõplikus praimerite kvaliteedihinnangus arvestatakse. Maskeerimisfunktsiooni sisselülitamisel arvestatakse automaatselt ka maskeeritud järjestusele praimerite disainimisel kehtivate piirangutega (vt peatükk 1.1.4, Kõressaar ja Remm (2007)).

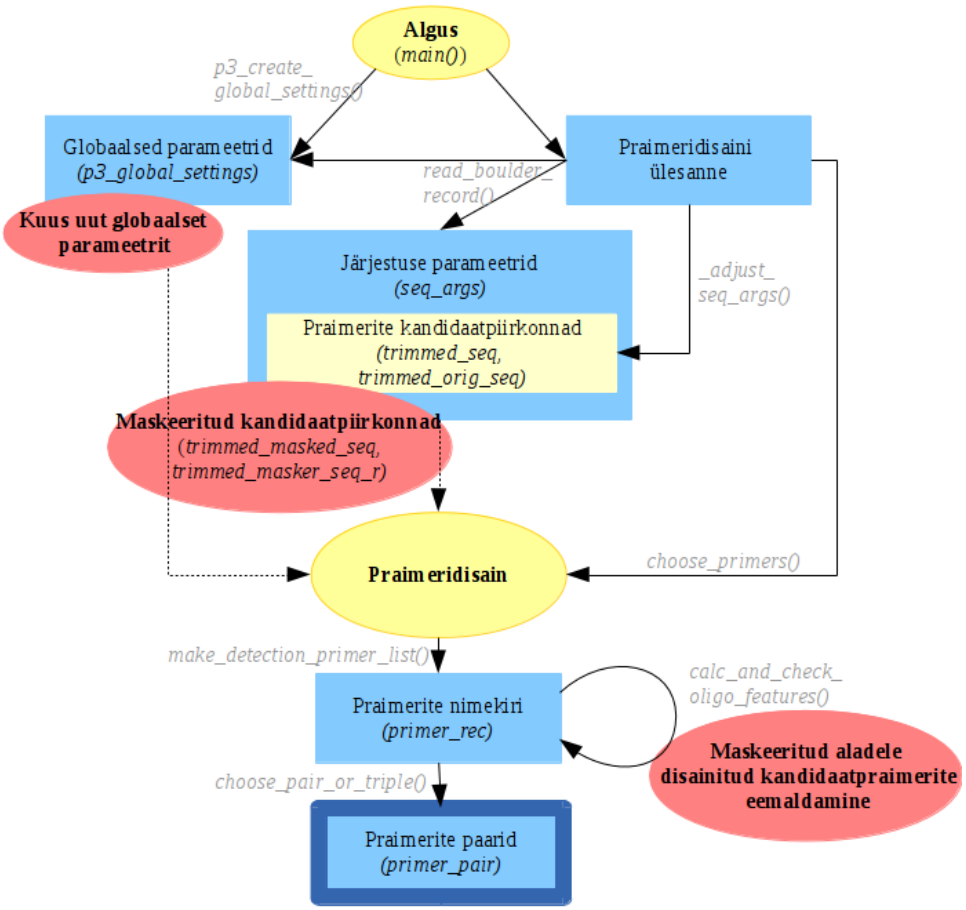
Tabel 1: Uued Primer3-e parameetrid

Parameeter	**Vaikeväärtus**	**Kirjeldus**
PRIMER_MASK_TEMPLATE	0	väärtuse 1 korral sisend-järjestus maskeeritakse
PRIMER_FAILURE_RATE	0,2	praimeri mittetöötamise tõenäosuse nivoo, mida maskeerimisalgoritm kasutab
PRIMER_MASK_5P_DIRECTION	1	praimeri 3'-otsast 5'-suunas maskeeritavate nukleotiidide arv
PRIMER_MASK_3P_DIRECTION	1	praimeri 3'-otsast 3'-suunas maskeeritavate nukleotiidide arv
PRIMER_MASKING_LIST_PREFIX	homo_sapiens	maskeerimiseks kasutatavate k-meeride tabeli nime eesliide
PRIMER_WT_FAILURE_RATE	0,0	kaal, millega praimeri mittetöötamise tõenäosust lõplikus pingereas arvestatakse

Primer3 programmi töö alguses loetakse kõik kasutaja antud globaalsed parameetrid mällu. Maskeerimisfunktsiooni sisselülitamisel loetakse mällu ka vajaminevad k-meeride tabelid. Seda tehakse programmi tööaja jooksul vaid nii mitu korda, kui tihti kasutaja tabeleid vahetada soovib.

Lähtudes kasutaja antud parameetritest, täpsustab Primer3-e algoritm esmalt lähtejärjestusel piirkonnad, kuhu praimerid disainitakse. Seejärel need piirkonnad maskeeritakse (vt ka joonis 6). Vaikimisi on praimeri mittetöötamise tõenäosuse lubatud ülempiiriks 0,2 ja maskeeritakse ainult üks nukleotiid praimeri 3'-otsast. Maskeeritud piirkonnad talletatakse Primer3-e järjestuse parameetrite struktuuris vaikimisi kahe järjestusena, üks kummagi ahela jaoks. Erandiks on olukorrad, kus kasutaja soovib disainida praimereid ainult ühele ahelale.

Lisaks ebasoovitud alade maskeerimisele leiab uuendatud Primer3-e algoritm seondumiskohtade arvust tingitud mittetöötamise tõenäosuse igale väljastatavale praimerile. Kasutaja saab määrata, kas ja kui suure kaaluga seda lõpliku praimerite pingerea koostamisel arvestatakse (vaikimisi seda ei arvestata).



Joonis 6: Primer3 programm integreeritud maskeerimisfunktsiooniga: *Maskeerimisalgoritmi integreerimiseks tuli defineerida kuus uut globaalset parameetrit (vt tabel 1). Uued funktsionaalsused lisati Primer3-e koodi kahte kohta (vt ka joonis 2). Esmalt praimerite seondumispiirkondade leidmisel täiustatud versioonis need piirkonnad maskeeritakse (trimmed_masked_seq). Lisaks arvutatakse iga kandidaatpraimeri kohta selle mittetöötamise tõenäosus (calc_and_check_oligo_features()), mida arvesse võttes saab väljastatud praimereid paremusjärjekorda panna.*

4 Tulemused

4.1 Maskeerimisalgoritmi funktsionaalsus

Käesoleva töö eesmärgiks oli lisada Primer3 programmi funktsionaalsus, mis takistaks praimerite disainimist sellistele sisendjärjestuse piirkondadele, millega sarnaseid lõike on uuritava organismi genoomis palju. Töö tulemusena valmis maskeerimisalgoritm, mis hindab potentsiaalsete seondumiskohtade arvust sõltuvat praimeri mittetöötamise tõenäosust ja maskeerib järjestuses need piirkonnad, kus saadud tõenäosus on kasutaja määratud nivoost suurem. Hinnangu leidmiseks kasutab algoritm logistilise regressiooni mudelit, mille kirjeldatavateks tunnusteks on vaadeldava praimeri 3'-otsa või sellega sarnaste k-meeride arvud.

Lisaks Primer3-e koodi integreerimisele implementeeriti maskeerimisalgoritm ka eraldi kompileeruva programmina. Saadud programm on Primer3-e versioonist paindlikum ja lubab kasutajal määrata rohkemate parameetrite väärtuseid. Maskeerimise programm võimaldab:

- kasutada praimerite mittetöötamise tõenäosuste hindamiseks käesoleva töö raames leitud mudelit (kasutatakse vaikimisi) või defineerida järjestuse maskeerimiseks uus k-meeride tabelitel põhinev mudel,
- anda mudeli parameetrid ette käsurealt või tekstifailina,
- määrata maskeerimise nivoo ehk lubatud praimeri mittetöötamise tõenäosuse ülempiir,
- maskeerida järjestust ilma mudelita, kasutades k-meeride sageduste väärtuseid ilma tõenäosusarvutuseta,
- maskeerida järjestust, kasutades erinevate genoomide/järjestuste pealt loodud k-meeride tabeleid,
- maskeerida praimeri 3'-otsa ümbruses vabalt valitud arv nukleotiide kummalegi poole,
- arvestada maskeerimisel järjestuse kumbagi ahelat eraldi või mõlemat koos,
- maskeerida järjestust vabalt valitud tähemärgiga või väiketähega,
- maskeerida järjestust, mis on FastA formaadis (sobib ka multi-FastA) ning saada väljundiks samamoodi formaaditud järjestus,

- kasutada maskeerimisalgoritmi tööjärjekordades (programm lubab kasutada standardsisendit ja -väljundit).

Primer3-e versioonis kasutatakse alati vaikemudelit. Maskeerimisel kasutatakse alati väiketähte ja maskeeritakse seda ahelat, kummale programm parasjagu praimerit disainida üritab. Kasutaja saab määrata maskeerimise nivoo, maskeeritavate nukleotiidide arvu ja k-meeride tabelid, mida maskeerimiseks kasutatakse.

4.2 Maskeerimisalgoritmi mudel

Üheks töö alamülesandeks oli leida statistiline mudel, mis hindaks praimeri mittetöötamise tõenäosust, lähtudes selle 3'-otsa k-meeride arvust genoomis. Sellist mudelit saab kasutada nii praimerite disainifaasis parimate praimerijärjestuste leidmiseks kui ka juba disainitud praimerite kvaliteedi hindamiseks. Potentsiaalsete mudelite komplekt loodi kasutades AIC-l põhineva valikukriteeriumiga ahnet algoritmi. Täiendavaks kriteeriumiks oli see, et nende kasutamisel ei vajaks maskeerimisalgoritm rohkem kui kahte erinevat k-meeride tabelit. Saadud mudelid jagunesid nelja gruppi vastavalt sellele, kas nende loomisel kasutati sõltumatute tunnuste ruutliikmeid ja/või uuritavast k-meerist kuni ühe ja kahe nukleotiidi võrra erinevate k-meeride sagedusi. Kokku genereeris algoritm 26 erinevat mudelit (vt Lisa B). Kõik saadud mudelid sisaldasid muutujaid, mis on arvutatavad 16-meeride sagedustest. Lisaks kasutasid mudelid k-meere pikkustega 6, 10, 11, 12 ja 14. Väljastatud mudelid olid grupiti väga sarnased, sest mitmete tunnuste väärtused olid paarikaupa väga tugevas korrelatsioonis, näiteks K11 ja K12 või ka K16_1MM ja K16_2MM. Seetõttu vaadeldi lähemalt vaid 5 mudelist koosnevat alamhulka, kusjuures eelistati võimalikult lihtsaid ja väikese AIC väärtusega mudeleid. Võrdlusesse lisati ka Andreson *et al.* (2008) mudelid GM1 ja GM1MM (vt tabel 2).

Tabel 2: Võrdluses olevad mudelid

Mudel	**Tunnused**
GM1	K16 + K15*K15 + K14
GM1MM	K15_2MM + K12_1MM*K12_1MM + K12
M1	K16*K16 + K16 + K16_2MM*K16_2MM + K16_2MM + K6_2MM + K6
M2	K16*K16 + K16 + K16_2MM*K16_2MM + K16_2MM + K6_2MM + K6_1MM
M3	K16*K16 + K16 + K16_1MM*K16_1MM + K16_1MM + K10*K10
M4	K16*K16 + K16 + K11*K11 + K11
M5	K16 + K11

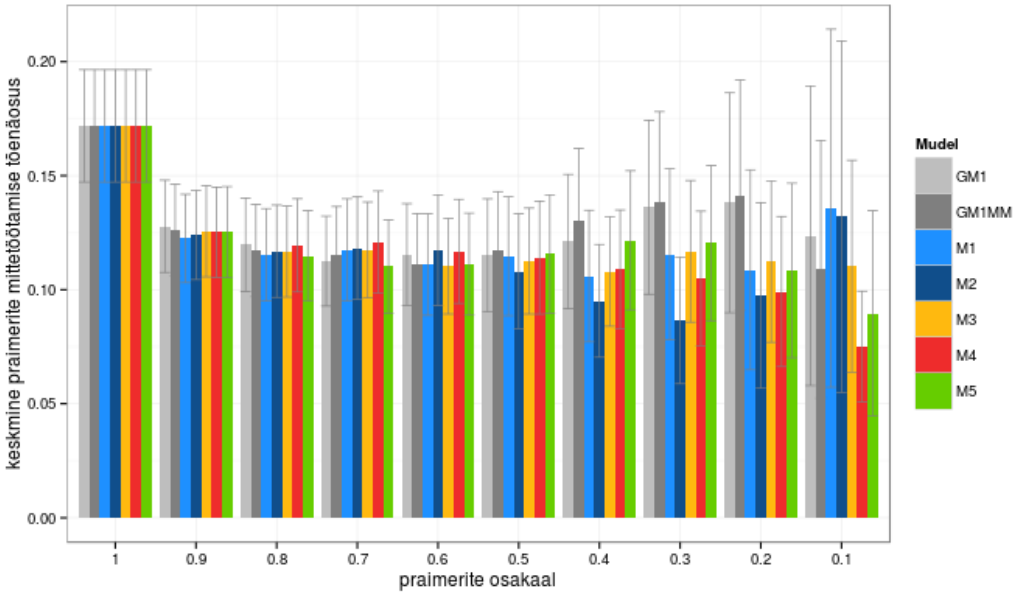
Primer3-s kasutatava mudeli valiku oluliseks kriteeriumiks oli see, kui kiiresti suudab maskeerimisalgoritm vaadeldavat mudelit kasutades järjestust analüüsida. Mõõdetud kiirused on toodud tabelis 3. Kõige kiiremini maskeerisid järjestust mudelid M4 ja M5. Võrreldes mudelitega, mis kasutavad oma tunnuste arvutamisel ka järjestuses oleva k-meeriga sarnaste k-meeride sagedusi (GM1MM, M1, M2 ja M3) on need kuni 40 korda kiiremad. Samuti on nad ligikaudu kaks korda kiiremad mudelist GM1, mis kasutab kolme erinevat k-meeride tabelit. Järjestuse ühe ja kahe ahela maskeerimise ajad ei erine teineteisest olulisel määral. Üheks põhjuseks on ilmselt see, et kõige pikemate k-meeride sagedusi, mille leidmine võtab kõige kauem aega, otsitakse kahe ahela puhul ikkagi ainult ühe korra.

Mudelite headust hinnati, kasutades sõltumatut testandmestikku. Esmalt leiti igale testandmestiku praimeripaarile kõigi võrdluses olevate mudelitega mittetöötamise tõenäosuse hinnangud. Seejärel eemaldati iga mudeli puhul andmestikust järjest suuremaid hinnangu järgi kehvemini töötavate praimerite osakaale ning jälgiti, kuidas muutub allesjäänud praimerite tegelike mittetöötamise tõenäosuste keskmine (vt joonis 7). Andmestikust kuni 50% kehvemate praimeripaaride eraldamisel käituvad kõik uuritavad mudelid väga sarnaselt. Rohkemate praimeripaaride eemaldamisel annavad parimaid tulemusi mudelid M4 ja M5. Nimetatud mudelite poolt 10% kõige paremaks hinnatud praimeripaaride keskmised tegelikud mittetöötamise tõenäosused on vastavalt 2,3 ja 1,9 korda väiksemad kui kõigi testandmestikus olevate praimeripaaride vastavad väärtused.

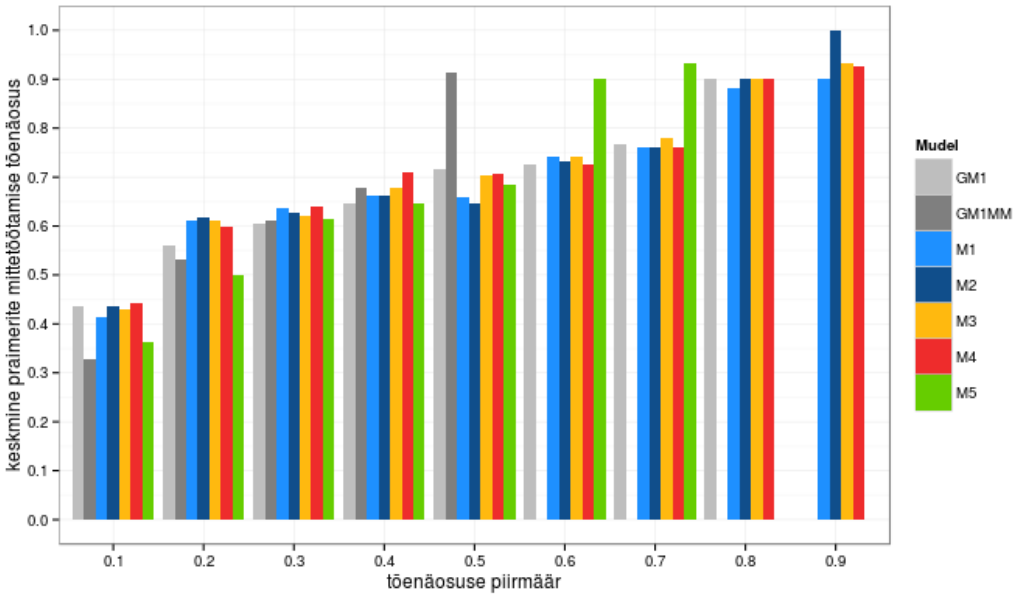
Lisaks vaadeldi seda, millised olid tegelike mittetöötamise tõenäosuste keskmised sellistel praimeripaaridel, mille mudeliga hinnatud mittetöötamise tõenäosus jäi mingist määratud nivoost kõrgemale (vt joonis 8). Tõenäosusnivoo $\alpha < 0{,}5$ korral on kõigi mudelite hinnangud taaskord sarnased. Suuremate nivoode puhul hindavad mudelid GM1 ja

***Tabel 3: Mudelite töökiirused**: Mõõdetud on maskeerimisalgoritmi töökiiruseid ühe ja kahe ahela maskeerimisel, kasutades erinevaid mudeleid. Kõik tabelis esitatud kiirused on arvutatud viie mõõtmise keskmistena. Ühikuteks on nukleotiidi sekundi kohta.*

Mudel	**Kiirus (üks ahel)**	**Kiirus (kaks ahelat)**
GM1	14 700	14 100
GM1MM	1 200	1 100
M1	800	800
M2	800	800
M3	2 700	2 700
M4	28 800	27 900
M5	32 200	27 100



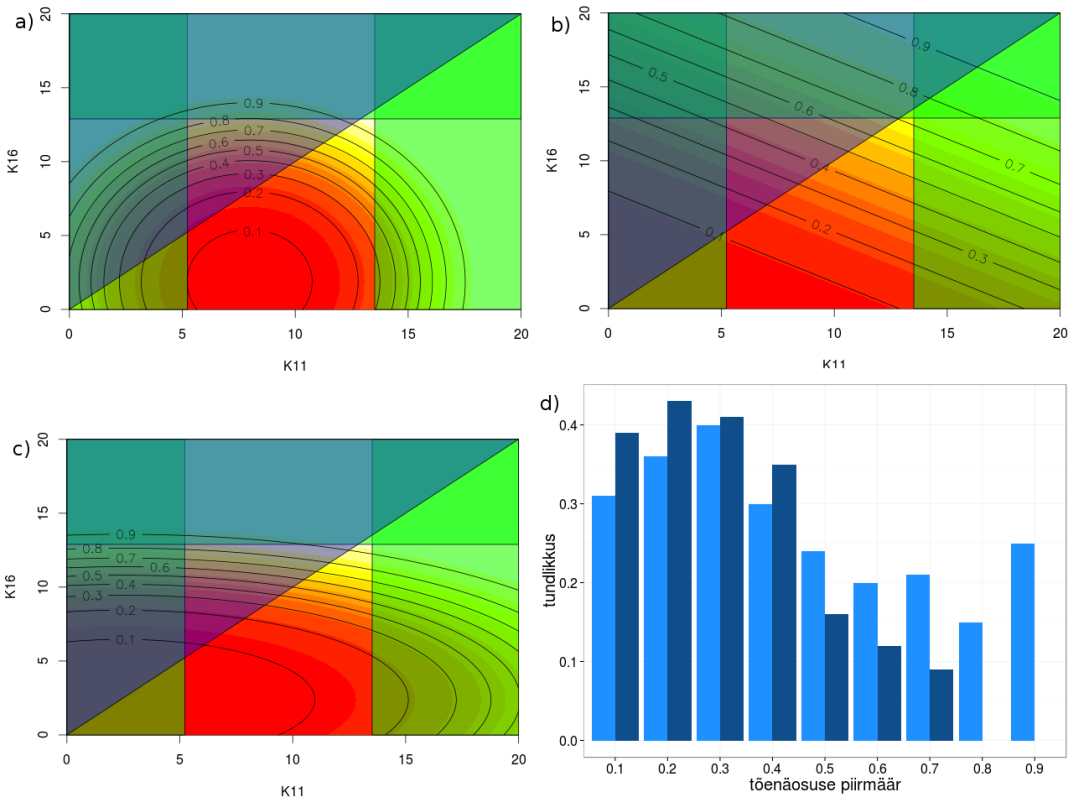
Joonis 7: Keskmised praimerite mittetöötamise tõenäosused: *Joonisel on kujutatud, kuidas muutub andmestikus olevate praimerite tegelike mittetöötamise tõenäosuste keskmine (y-telg), kui alles jätta vaid fikseeritud osakaal (x-telg) väiksema hinnatud mittetöötamise tõenäosusega praimereid. Hinnanguid leiti seitsme erineva mudeliga.*



Joonis 8: Maskeeritud praimerite tegelikud mittetöötamise tõenäosused erinevate mudelite ja maskeerimisnivoode korral: *Tulpade puudumine tähendab, et mudel ei maskeerinud antud nivoo juures ühtegi praimerit.*

M5 praimeripaaride mittetöötamise tõenäosuseid tegelikest veidi madalamateks ja mudel GM1MM lausa kuni 2 korda madalamaks.

Lähemalt vaadeldi kõige lihtsamaid mudeleid M4 ja M5, mis võimaldavad ka kõige kiiremaid arvutusi (vt jooniseid 9a ja 9b). Selleks hinnati mudelite koefitsiendid kõigi 1313 praimeripaari pealt. Saadud mudelite kujudest on näha, et mõlemad töötavad andmestikus olevate vaatluste piirkonnas sarnaselt, kuigi M5 tõenäosuste hinnangud on M4



Joonis 9: Mudelite M4 ja M5 võrdlus: *Joonised a), b) ja c) tähistavad vastavalt mudelite M4, M5 ja M4mod (eraldi hinnatud vabaliikmega mudel) visualisatsioone muutujate väärtuste piirkonnas* $[0, 20] \times [0, 20]$. *K11 ja K16 on praimerite 3'-otstes olevate* 11- *ja* 16*-meeride logaritmitud sagedused. Puna-kollane gradient tähistab mudeli hinnatud mittetöötamise tõenäosust, kusjuures punane märgib madalamat tõenäosust. Rohekad alad tähistavad tunnuste väärtuseid, mis jäävad mudeli treeningandmestiku piirkonnast välja. Sinine ala tähistab väärtuste komplekte, mida reaalsuses kunagi esineda ei saa (*16*-meeride sagedus pole kunagi väiksem kui* 11*-meeride sagedus). On näha, et mudel M4 hindab väikeste K11 väärtuste korral praimeri mittetöötamise tõenäosust väga suureks. Mudelid M5 ja M4mod on bioloogilises mõttes relevantsemad. Joonisel d) on võrreldud mudelite M4mod ja M5 tundlikkust eri tõenäosusnivoode korral. Mudel M5 töötab paremini madalamate ja M4mod kõrgemate tõenäosusnivoodega.*

omadest veidi madalamad. Seevastu väljaspool vaatluste piirkonda hindab M4 mittetöötamise tõenäosuseid väga suurteks. PCR-i metoodikat aluseks võttes võime öelda, et mudeli M4 hinnangud piirkonnas K11= $[0; 5{,}22]$ ei ole bioloogiliselt korrektsed. Mudeli parandamiseks leiti sellise vabaliikme väärtus, mis tagaks mudelile nimetatud piirkonnas õige kuju. Seejärel kasutati leitud vabaliikme väärtust mudeli ülejäänud koefitsientide hindamiseks. Saadud mudel M4mod on oma kujult õigem (vt joonis 9c).

Mudelitest M4mod ja M5 parema valimiseks, vaadeldi mudelite tundlikkust (ingl. k. *sensitivity*) erinevate maskeerimisnivoode α korral (vt joonis 9d). Saadud tulemustest on näha, et $\alpha < 0{,}5$ korral leiab mudel M5 rohkem õigeid positiivseid, $\alpha \geq 0{,}5$ korral on parem mudel M4mod. Et kasutajat huvitavad eelkõige madalate mittetöötamise tõenäosustega praimerid, valiti parimaks mudeliks M5. Mudeli kuju koos kogu andmestiku pealt hinnatud parameetritega on järgmine:

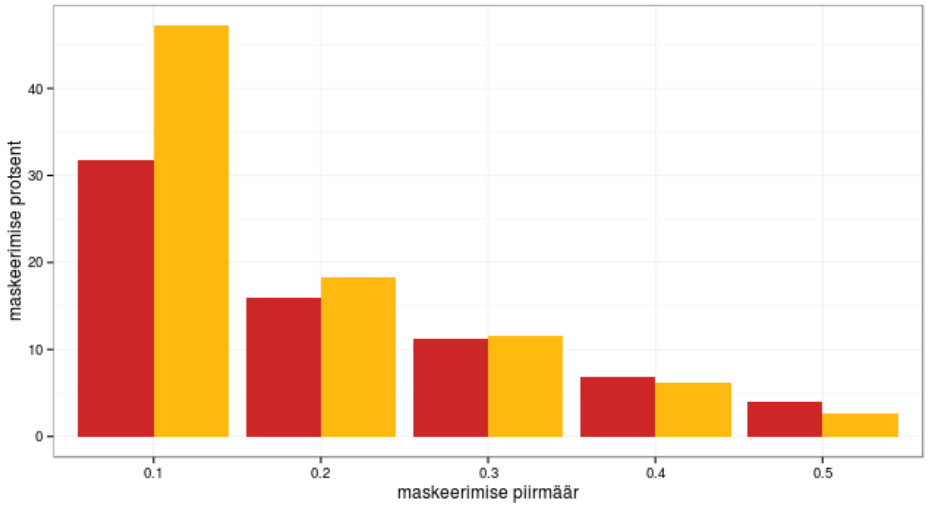
$$p_{\text{mittetöötamine}} = \frac{e^{-4{,}336+0{,}1772*\text{K11}+0{,}239*\text{K16}}}{1+e^{-4{,}336+0{,}1772*\text{K11}+0{,}239*\text{K16}}}.$$

Sooviti teada, kui palju valitud mudel eri tõenäosusnivoode korral lähtejärjestusest keskmiselt maskeerib. Selleks kasutati sadat inimese 1. kromosoomist juhuslikult võetud 1000-nukleotiidset järjestust ja inimese Y kromosoomi, mis sisaldab enam kui 50% ulatuses erinevaid kordusjärjestusi (Smith *et al.*, 1987). Maskeerimisprotsente on kujutatud joonisel 10. Tõenäosusnivood 0,1 kasutades, maskeeritakse Y kromosoomis ligikaudu pooled ja juhuslikult valitud genoomijärjestustes kolmandik nukleotiididest. Kõrgemate piirmäärade korral muutuvad maskeeritud järjestuse osakaalud kahe andmestiku vahel sarnasemaks.

4.3 Maskeerimisalgoritmi aja- ja mälukasutus

Maskeerimisalgoritmi ajakasutus sõltub eelkõige maskeeritava järjestuse pikkusest ja k-meeride tabelitest, mida mudeli tunnuste väärtuste arvutamiseks kasutatakse. Programmi tööaeg pikeneb järjestuse pikkuse kasvades lineaarselt ja k-meeride tabeli suuruse kasvades logaritmiliselt. Tabelid on suuremad pikemate k-meeride korral. Mudeliga, mis sisaldab ainult tunnust K6, on maskeerimise kiirus kuni ligikaudu 640 000 nukleotiidi sekundis, K16-ga mudeli korral on kiiruseks ligikaudu 40 000 nukleotiidi sekundis. Lisaks on algoritmi tööajale oluline mõju sellel, kas mudelid sisaldavad tunnuseid, mille arvutamiseks kasutatakse lisaks järjestuses olevate k-meeride sagedustele ka neist ühe või kahe nukleotiidi võrra erinevate k-meeride sagedusi (vt tabel 4).

Maskeerimisalgoritmi mälukasutus sõltub kasutatavate k-meeride tabelite suurusest ja maskeeritava järjestuse pikkusest. Kuna tabelite poole pöördutakse *memory mapping* funktsiooni abil, siis loetakse operatiivmällu vaid see osa failidest, mida reaalselt kasutati. Näiteks 100 000 nukleotiidi pikkuse järjestuse maskeerimine mudelitega,



Joonis 10: Maskeeritud järjestuste osakaalud: *Joonisel on kujutatud eri piirnivoode ($\alpha \leq 0{,}5$) korral maskeeritud järjestuse protsenti juhuslikult valitud inimese genoomi piirkondade (punane) ja Y kromosoomi (kollane) korral. Y kromosoomi puhul on arvestatud ainult neid positsioone, millel asuv nukleotiid on algses järjestuses teada (pole N).*

mis sisaldavad ainult 6-meeride sagedusi, kasutati 1MB operatiivmälu, 12-meeride ja 16-meeride puhul on need arvud vastavalt 44MB ja 2GB.

Vaikemudeli väärtuste arvutamiseks kasutatakse 11- ja 16-meeride tabeleid. Nende suurused on vastavalt 24MB (2 097 100 unikaalset 11-meeri) ja 9GB (799 095 062 unikaalset 16-meeri). Maskeerimine selle mudeliga kasutab 100 000-nukleotiidse järjestuse puhul maksimaalselt 9GB operatiivmälu. Maskeerimise kiirus on ligikaudu 32 000 nukleotiidi sekundis.

Tabel 4: Mudeli töökiiruse sõltuvus kasutatavatest tunnustest*: Maskeerimisalgoritmi kiiruste mõõtmiseks kasutati kolme erineva pikkusega k-meeride sagedusi ($N = \{6, 12, 16\}$). Võrreldi mudeleid, mille tunnuseks olid ainult järjestuses olevate k-meeride sagedused (KN) ning mudeleid, mis arvestasid neist k-meeridest kuni ühe (KN_1MM) ja kuni kahe (KN_2MM) nukleotiidse erinevusega k-meeride sagedusi. Viimaste töökiirused olid 6-meeride puhul esimesest vastavalt 2 ja 5 korda aeglasemad, 16-meeride puhul aga lausa 12 ja 67 korda aeglasemad.*

N	KN	KN_1MM	KN_2MM
6	641 000	316 500	125 000
12	191 600	27 700	7 200
16	40 000	3 300	600

4.4 Maskeerimisalgoritmi kasutamine Primer3-e koosseisus

Maskeerimisalgoritmi vajalikkuse testimiseks praimeridisaini kontekstis kasutati Primer3e sisendjärjestustena sadat inimese 1. kromosoomi järjestust, millest igaüks oli 1000 nukleotiidi pikk. Igale etteantud järjestusele disainiti praimereid, kasutades Primer3-e parameetrite vaikeväärtuseid, kuid mitte maskeerimist. Seejärel disainiti samadele järjestustele praimereid nii, et maskeerimisfunktsioon oli sisse lülitatud ja maskeerimisnivooks oli valitud 0,2. Ühtlasi arvestati praimeri mittetöötamise tõenäosust ka kandidaatpraimerite pingeritta seadmisel kaaluga 1,0. Mõlemal juhul lasti Primer3-l väljastada iga järjestuse jaoks üks parim praimeripaar, kokku 200 praimerit.

Saadud praimeritele leiti nende mittetöötamise tõenäosused. Maskeerimata järjestuste korral oli disainitud praimerite keskmine mittetöötamise tõenäosus 0,14. Kõigist praimeritest 47-l oli nimetatud tõenäosus üle 0,25 ja 13-l lausa üle 0,5. Eelnevalt järjestusi maskeerides oli keskmiseks disainitud praimerite mittetöötamise tõenäosuseks 0,069 ning ühelgi praimeril polnud see üle etteantud piirmäära 0,2. Seega võimaldab maskeerimise kasutamine disainida keskmiselt kaks korda paremaid praimerijärjestusi.

Lisaks vaadati seda, kui palju praimerikandidaate eemaldati valikust just maskeerimise tõttu. Iga disainitud praimeripaari kohta eemaldati maskeerides keskmiselt 16% kandidaatidest. Maksimaalne eemaldatud praimerite osakaal oli aga lausa 96%.

5 Arutelu

PCR-i praimerite töökindlus sõltub lisaks muudele parameetritele ka vaadeldava praimeri seondumiskohtade arvust genoomis (Ghedira *et al.*, 2009). Paljude seondumiskohtadega praimerid tekitavad reaktsiooni käigus suurema tõenäosusega valeprodukte või ei tööta üldse. Käesoleva töö raames koostati praimeri seondumiskohtade arvust sõltuva mittetöötamise tõenäosuse mudel, mis integreeriti maskeerimisalgoritmi, mille funktsionaalsus omakorda lisati ühe enamkasutatava praimeridisaini programmi Primer3 loogikasse.

Praimerite mittetöötamise mudeli valimisel seati kriteeriumiks, et selle tunnuste arvutamine ei tohi kasutada rohkem kui kahte k-meeride tabelit. Seda eelkõige põhjusel, et kasutajal poleks vaja eelnevalt genereerida väga paljusid tabeleid ning et Primer3-ga kaasapandavad tabelid ei võtaks palju kõvaketta ruumi. Samal põhjusel otsustati maksimaalset k-meeri pikkust piirata 16-ga (näiteks inimese 18-meeride korral on k-meeride tabeli suurus juba 21GB). Pikemate k-meeride mudelisse kaasamist testiti, kuid see ei andnud oluliselt paremaid tulemusi, sest praimerite 3'-otstes olevate sarnaste pikkustega k-meeride sagedused on väga tugevas omavahelises korrelatsioonis (näiteks 16- ja 18-meeride omavaheline korrelatsioon on ligikaudu 0,95).

Varasemalt olid publitseeritud Andreson *et al.* (2008) mudelid GM1 ja GM1MM. Neid aga otsustati maskeerimisalgoritmis vaikemudelitena mitte kasutada, sest need ei sisaldanud tunnustena ainult k-meeride arvudest tuletatavaid väärtuseid, vaid ka näiteks praimeri GC-sisaldust. Samuti kasutab mudel GM1 kolme k-meeride tabelit, mis ei vasta antud töös seatud kriteeriumitele. GM1MM-i ja teiste sarnaste mudelite, mis arvestavad oma tunnuste väärtuste leidmisel ka järjestuses olevate k-meeridega sarnaste k-meeride sagedusi, kasutamist ei soosi eelkõige nende pikem tööaeg. Seetõttu valiti vaikemudel kõige lihtsamate ja kiiremate mudelite hulgast, otsustades sellise mudeli kasuks, mis töötab paremini madalamate mittetöömise tõenäosusnivoode juures. Ülejäänud mudelitega maskeerimiseks saab kasutaja valitud mudeli programmile ise ette anda.

Kuigi käesolevas töös kasutati mudeli loomisel inimese genoomis olevate k-meeride sagedusi, siis eeldatavasti sobib sama mudel ka teiste enam-vähem sama suurte eukarüootsete genoomide (enamik imetajaid) maskeerimiseks. Seda eelkõige põhjusel, et sarnaste suurustega genoomide k-meeride sagedusjaotused on sarnased. Ka juhul, kui uuritav genoom on palju suurem, kuid seda eelkõige korduvate motiivide või järjestuste arvelt (näiteks mitmete taimede genoomid), võivad töös leitud mudelid hästi sobida, kui suurem osa k-meeride sagedustest on ligikaudu samad. Lisaks võib eeldada, et isegi kui suuremate genoomide puhul ei ole mudeli ennustatud praimerite mittetöötamise tõenäosused õiged, on ennustus õige suunaga. See tähendab, et mistahes ennustatud tõenäosuse nivoo järgi maskeerides välditakse alati kõige halvemate praimerite disainimist.

Primer3-s kasutatakse implementeeritud funktsioone kahes erinevas praimeridisaini etapis. Sisendjärjestuse maskeerimine teostatakse enne potentsiaalsete praimerikandidaatide valimist. See võimaldab kiiresti eemaldada hulga kõige kehvemaid praimereid. Nendele praimeritele, mis alles jäävad ja mille hulgast Primer3 sobivaid praimeripaare otsib, arvutatakse disaini hilisemas järgus mittetöötamise tõenäosus, mille abil saab leitud praimeripaare pingeritta seada. See võimaldab allesjäänud praimerite hulgast välja selekteerida kõige paremad. Taoline algoritmi ülesehitus tagab selle, et Primer3 väljastaks võimalikult head praimerid ning ei väljastaks kunagi neid praimereid, mis on mingist kasutaja etteantud nivoost kehvemad. See tuleb kasuks olukordades, kus häid praimereid on väga raske disainida. Sellisel juhul saab kasutaja maskeerimisnivoo panna kõrgeks ning leida saadud kehvade praimerite hulgast ikkagi kõige paremad.

Kasutades algoritmi ainult praimerite pingerea leidmiseks, langetatakse praimeri sobivuse või ebasobivuse otsus ainult selle ühe praimeri järjestust arvestades. Samas võib kasutaja tahta vaatluse alt eemaldada ka kehva praimeri vahetus naabruses olevad praimerid. Seda võimaldab vaid järjestuse maskeerimine. Seal saab kasutaja määrata, kui palju nukleotiide iga mittesobiva praimeri seondumiskoha ümbrusest maskeeritakse.

Primer3 lubab juba varasemates versioonides sisendina ette anda väiketähtedega maskeeritud järjestuse. Ka uue maskeerimisfunktsionaalsusega on järjestust võimalik maskeerida enne praimeridisaini protsessi. See võimaldab maskeeritud järjestust kasutada korduvalt. Lisaks annab eraldiseisev maskeerimise programm kasutajale suurema vabaduse parameetrite väärtuseid valida. Integreeritud maskeerimisfunktsioon võimaldab aga sisendjärjestust vaadelda just käesoleva disainiülesande kontekstis. Näiteks tagab programm, et maskeeritud saaks õige järjestuse ahel. Primer3-e sisendiks saab anda vaid ühe maskeeritud järjestuse ning mõlema ahela praimerid disainitakse seda kasutades.

Primer3 programmi koos ja ilma maskeerimisfunktsioonita kasutades on näha, et maskeerimine vähendab saadud praimerite ennustatud mittetöötamise tõenäosuseid ligikaudu kaks korda. Olulisem aga on täheldada, et ilma maskeerimata järjestusele disainitakse ka selliseid praimereid, mille mittetöötamise tõenäosus on väga suur (ehk neil on väga palju erinevaid seondumiskohti uuritavas genoomis) ning maskeerimine võimaldab need kandidaadid juba varakult disainiprotsessist eemaldada. Käesoleva uurimuse viimase sammuna tuleks disainida praimereid nii maskeeritud kui ka maskeerimata järjestusele ning nende töötamise edukust laboritingimustes testida.

Lisaks suure mittetöötamise tõenäosusega praimerite väljaselekteerimisele, on maskeerimisalgoritmil ka teisi potentsiaalseid kasutusvõimalusi. Üheks selliseks on liigispetsiifiliste praimerite disainimine. Selleks tuleb kasutajal uuritava liigi paljundatav genoomipiirkond maskeerida, kasutades teiste liikide genoomide k-meeride ühendtabelit ja k-meeride sagedusnivood 1. Sellisel juhul jäävad maskeerimata ainult need alad, mis on

uuritavale liigile spetsiifilised. Taolisele järjestusele saab disainida praimereid, mis teoreetiliselt ei tohiks seonduda ühegi teise liigi genoomile. Sellist lähenemist saab kasutada ka näiteks bakterispetsiifiliste praimerite disainiks olukorras, kus proov sisaldab inimese DNA-d (näiteks meditsiinilises diagnostikas). Sel juhul maskeeritakse uuritav järjestus lähtuvalt inimese k-meeride tabelitest ja sellega välditakse selliste praimerite disainimist, mis ei töötaks nende inimese genoomsele DNA-le seondumise tõttu, ehkki bakteri genoomis on nende järjestused unikaalsed.

Kokkuvõte

Polümeraasi ahelreaktsioon (PCR) on üks enimkasutatavaid biotehnoloogilisi meetodeid nukleotiidsete järjestuste paljundamiseks. Reaktsiooni õnnestumine sõltub paljudest teguritest, millest mitmed on seotud PCR-i praimerite järjestuste omadustega. Praimerid peavad olema disainitud selliselt, et nad seonduks paljundatava DNA piirkonnaga stabiilselt ja võimalikult spetsiifiliselt. Praimerid, mis seonduvad uuritava genoomi paljude piirkondadega, põhjustavad valeproduktide tekke või ei tööta üldse.

Käesoleva töö esimeseks eesmärgiks oli koostada statistiline mudel, mille abil hinnata praimerite mittetöötamise tõenäosuseid. Kirjeldavate tunnustena kasutati disainitava praimeri 3'-otsas asuvate k-meeride arve uuritavas genoomis. Mudelisse tunnuste valimisel oli oluliseks kriteeriumiks, et nende väärtuste arvutamiseks ei oleks vaja rohkem kui kahe erineva pikkusega k-meere. Parimaks osutus mudel, mis sisaldas tunnustena logaritmitud 11 ja 16-meeride arve.

Teiseks eesmärgiks oli implementeerida algoritm, mis võtaks sisendiks lähtejärjestuse ja praimerite mittetöötamise tõenäosust hindava mudeli ning maskeeriks järjestuses need piirkonnad, millele disainitud praimerite mittetöötamise tõenäosused on kasutaja määratud maksimaalsest lubatud tõenäosusest suuremad. Töö käigus valmis käsurealt käivitatav programm, mis võimaldab järjestust maskeerida kasutaja vabalt valitud mudeli ja piirnivooga. Samuti saab kasutaja programmi käivitades määrata maskeerimise ranguse ja ulatuse ning selle, kumba järjestuse ahelat ja millise tähemärgiga maskeeritakse.

Töö kolmandaks eesmärgiks oli loodud maskeerimisfunktsiooni integreerimine praimeridisaini programmi Primer3. Uut funktsionaalsust lisati disainiprotsessi kahte erinevasse järku. Esmalt võimaldati sisendjärjestuse maskeerimine enne kandidaatpraimerite genereerimist. Selle abil saab vältida paljude seondumiskohtadega praimerite disainimist. Lisaks arvutab täiendatud Primer3 igale kandidaatpraimerile tema mittetöötamise tõenäosuse. Seda saab arvestada väljastatud praimerite pingeritta seadmisel.

Viimaks testiti, kas ja kui palju maskeerimise kasutamine disainitud praimerite mittetöötamise tõenäosust vähendas. Tulemuseks saadi, et maskeeritud järjestustelt disainitud praimerite keskmine mittetöötamise tõenäosus oli kaks korda madalam kui maskeerimata järjestuste puhul. Lisaks tagab maskeerimine, et ükski väljastatud praimer pole kehvema mittetöötamise tõenäosusega kui see, mida kasutaja maksimaalselt lubab.

Viited

Altschul, S. F., Madden, T. L., Sch ? affer, A. A., Zhang, J., Zhang, Z., Miller, W., Lipman, D. J. (1997). Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucleic Acids Res. 25: 3389-402.

Andreson, R., Reppo, E., Kaplinski, L., Remm, M. (2006). GENOMEMASKER package for designing unique genomic PCR primers. BMC Bioinformatics. 7:172.

Andreson, R., Möls, T., Remm, M. (2008). Predicting failure rate of PCR in large genomes. Nucleic Acids Res. 36:e66.

Chuang, L. Y., Cheng, Y. H., Yang, C. H. (2013). Specific primer design for the polymerase chain reaction. Biotechnol Lett. 35:1541-9.

Daly, M. J., Lincoln S. E., and Lander E. S. (1991). „PRIMER", Whitehead Institute/MIT Center for Genome Research.

Dawson, E., Abecasis, G. R., Bumpstead, S., Chen, Y., Hunt, S., Beare, D. M., Pabial, J., Dibling, T., Tinsley, E., Kirby, S., Carter, D., Papaspyridonos, M., Livingstone, S., Ganske, R., Lõhmussaar, E., Zernant, J., Tõnisson, N., Remm, M., Mägi, R., Puurand, T., Vilo, J., Kurg, A., Rice, K., Deloukas, P., Mott, R., Metspalu, A., Bentley, D. R., Cardon, L. R., Dunham, I. (2002). A first-generation linkage disequilibrium map of human chromosome 22. Nature. 418:544-8.

Deorowicz, S., Kokot, M., Grabowski, S. (2014). KMC 2: Fast and resource-frugal k-mer counting. 1:1-21.

Duvanenko, V. J. (2009). In-place Hybrid N-bit-Radix Sort. Algorithm Improvement through Performance Measurement: Part 3. Dr. Dobb's Journal.

Gervais, A. L., Marques, M., Gaudreau, L. (2010). PCRTiler: automated design of tiled and specific PCR primer pairs. Nucleic Acids Research, 38:W308?12.

Ghedira, R., Papazova, N., Vuylsteke, M., Ruttink, T., Taverniers, I., De Loose, M. (2009). Assessment of primer/template mismatch effects on real-time PCR amplification of target taxa for GMO quantification. J Agric Food Chem. 57:9370-7.

Kaplinski, L., Lepamets, M., Remm, M. (2015). GenomeTester4: a toolkit for performing basic set operations - union intersection and complement on k-mer lists. Gigascience. *Vastu võetud.*

Burnham, K. P., Anderson, D. R. (2002). Model Selection and Multimodel Inference: A Practical Information-Theoretic Approach. Springer. II trükk. lk 60-64,70.

Kent, W. J. In-silico PCR. http://genome.ucsc.edu/

Kent, W. J. (2002). BLAT - the BLAST-like alignment tool. Genome Res. 12:656-64.

Kernighan, B. W., Ritchie, D. M. 1988. The C programming language. 2nd ed., Prentice Hall PTR, Englewood Cliffs, New Jersey.

Kramer, M. F., Coen, D. M. (2001). Enzymatic amplification of DNA by PCR: standard procedures and optimization. Curr Protoc Mol Biol. ptk 15: osa 15.1.

Kuhn, A., Ong, Y. M., Quake, S. R., Burkholder, W. F. (2015). Read count-based method for high-throughput allelic genotyping of transposable elements and structural variants. BMC Genomics. 16:508.

Kõressaar, T., Remm, M. (2007). Enhancements and modifications of primer design program Primer3. Bioinformatics 23:1289-1291.

Lepamets, M. (2014). Oligomeeridel põhinevate bioinformaatiliste meetodite kasutamine bakterite määramiseks sekveneerimislugemitest. TÜ MRI bakalaureusetöö, juhendaja Lauris Kaplinski.

Marçais, G., Kingsford, C. (2011). A fast, lock-free approach for efficient parallel counting of occurrences of k-mers. Bioinformatics 27:764-770.

Mullis, K., Faloona, F., Scharf, S., Saiki, R., Horn, G., Erlich, H. (1984). Specific enzymatic amplification of DNA in vitro: the polymerase chain reaction. Cold Spring Harb Symp Quant Biol. 51 Pt 1:263-73.

Ning, Z., Cox, A. J., Mullikin, J. C. (2001). SSAHA: a fast search method for large DNA databases. Genome research 10:1725-9.

Onodera, K., Melcher, U. (2004). Selection for 3' end triplets for polymerase chain reaction primers. Mol Cell Probes. 18:369-72.

Piriyapongsa, J., Ngamphiw, C., Assawamakin, A., Wangkumhang, P., Suwannasri, P., Ruangrit, U., Agavatpanitch, G., Tongsima, S. (2009). RExPrimer: an integrated primer designing tool increases PCR effectiveness by avoiding 3' SNP-in-primer and mispriming from structural variation. BMC Genomics. 3:S4.

PREMIER Biosoft. PremierPrimer. http://www.premierbiosoft.com/primerdesign

PrimerDigital. FastPCR. http://www.primerdigital.com/fastpcr.html

RepeatMasker. http://www.repeatmasker.org

Rizk, G., Lavenier, D., Chikhi, R. (2013). DSK: K-mer counting with very low memory usage. Bioinformatics 29:652-653.

Rozen, S., Skaletsky, H. (2000). Primer3 on the WWW for general users and for biologist programmers. Methods Mol Biol. 132:365-86.

Rychlik, W. (2007) OLIGO 7 Primer Analysis Software, in Methods in Molecular Biology Vol. 402: PCR Primer Design; Ed. A. Yuryev; Humana Press Inc., Totowa, NJ. lk. 35-59.

SantaLucia, J. Jr., Hicks, D. (2004). The thermodynamics of DNA structural motifs. Annu Rev Biophys Biomol Struct. 33:415-40.

Sipos, R., Székely, A. J., Palatinszky, M., Révész, S., Márialigeti, K., Nikolausz, M. (2007). Effect of primer mismatch, annealing temperature and PCR cycle number on 16S rRNA gene-targetting bacterial community analysis. FEMS Microbiol Ecol. 60:341-50.

Smith, K. D., Young, K. E., Talbot, C. C. Jr, Schmeckpeper, B. J. (1987). Repeated DNA of the human Y chromosome. Development. 101:77-92.

Stratmann, S. A., van Oijen, A. M. (2014). DNA replication at the single-molecule level. Chem Soc Rev. 43:1201-20.

Thompson, R., Zoppis, S., McCord, B. (2012). An overview of DNA typing methods for human identification: past, present, and future. Methods Mol Biol. 830:3-16.

Untergasser, A., Nijveen, H., Rao, X., Bisseling, T., Geurts, R., Leunissen, J. A. (2007). Primer3Plus, an enhanced web interface to Primer3. Nucleic Acids Res. 35:W71-4.

Untergasser, A., Cutcutache, I., Koressaar, T., Ye, J., Faircloth, B. C., Remm, M., Rozen, S. G. (2012). Primer3–new capabilities and interfaces. Nucleic Acids Res. 40:e115.

Vallone, P. M., Butler, J. M. (2004). AutoDimer: a screening tool for primer-dimer and hairpin structures. Biotechniques. 37:226-31.

Valones, M. A. A., Guimarães, R. L., Brandão, L. A. C., Souza, P. R. E. de, Carvalho, A. de A. T., Crovela, S. (2009). Principles and applications of polymerase chain reaction in medical diagnostic fields: a review. Brazilian Journal of Microbiology, 40:1-11.

Wu, J.-S., Lee, C., Wu, C.-C., Shiue, Y.-L. (2004). Primer design using genetic algorithm. Bioinformatics 20:1710-17.

Ye, J., Coulouris, G., Zaretskaya, I., Cutcutache, I., Rozen, S., Madden, T. L. (2012). Primer-BLAST: a tool to design target-specific primers for polymerase chain reaction. BMC Bioinformatics. 13:134.

Lisad

Lisa A: Praimerite andmestiku näidis

Kokku sisaldab andmestik 1313 praimeripaari identifikaatorit. Iga paari kohta on toodud õnnestunud ja ebaõnnestunud PCR-i reaktsioonide arvud ning 33 logaritmitud k-meeride sagedustel põhinevat tunnust.

ID	POS_RESULTS	NEG_RESULTS	K6	K6_1MM	K6_2MM
10002	154	16	14.915	17.685	19.556
10003	157	13	14.194	17.031	19.023
10005	142	6	14.513	17.414	19.342
10007	159	11	14.752	17.455	19.444
10009	139	9	14.026	17.14	19.321
10010	90	58	14.535	17.272	19.274
10011	138	10	14.431	17.244	19.253
10012	110	38	15.024	17.703	19.613
10014	134	14	14.32	17.047	19.123
10015	138	10	14.33	17.322	19.292
10016	138	14	14.396	17.125	19.123
10020	146	6	14.775	17.52	19.467
10022	87	65	15.032	17.581	19.417
10023	146	6	14.806	17.296	19.284
10026	147	5	14.231	17.219	19.314
10027	169	4	14.684	17.32	19.191
10028	146	6	14.599	17.225	19.145
10029	141	11	14.32	16.905	19.045
10030	136	16	14.325	17.259	19.353
10032	152	3	14.355	17.327	19.305

Lisa B: AIC-l põhineva algoritmiga genereeritud mudelid

mudeli number	tunnuste loetelu	AIC
1	K16*K16 K16 K16_2MM*K16_2MM K16_2MM K6_2MM K6	5020.5
2	K16*K16 K16 K16_2MM*K16_2MM K16_2MM K6_2MM K6_1MM	5020.32
3	K16*K16 K16 K16_2MM*K16_2MM K16_2MM K6_2MM K6*K6	5020.46
4	K16*K16 K16 K16_2MM*K16_2MM K16_2MM K6_2MM K6_1MM*K6_1MM	5020.3
5	K16*K16 K16 K16_2MM*K16_2MM K16_2MM K10 K10_2MM	5027.3
6	K16*K16 K16 K16_2MM*K16_2MM K16_2MM K10 K10_2MM*K10_2MM	5027.42
7	K16*K16 K16 K16_2MM*K16_2MM K16_2MM K6_2MM*K6_2MM K6	5020.52
8	K16*K16 K16 K16_2MM*K16_2MM K16_2MM K6_2MM*K6_2MM K6_1MM	5020.32
9	K16*K16 K16 K16_2MM*K16_2MM K16_2MM K6_2MM*K6_2MM K6*K6	5020.48
10	K16*K16 K16 K16_2MM*K16_2MM K16_2MM K6_2MM*K6_2MM K6_1MM*K6_1MM	5020.3
11	K16*K16 K16 K16_2MM*K16_2MM K16_2MM K10*K10 K10_2MM	5026.92
12	K16*K16 K16 K16_2MM*K16_2MM K16_2MM K10*K10 K10_2MM*K10_2MM	5027.02
13	K16*K16 K16 K16_2MM*K16_2MM K16_2MM	5028.68
14	K16*K16 K16 K16_1MM*K16_1MM K16_1MM K10 K10*K10	5032.22
15	K16*K16 K16 K16_1MM*K16_1MM K16_1MM K14_1MM	5031.8
16	K16*K16 K16 K16_1MM*K16_1MM K16_1MM K10*K10	5031.68
17	K16*K16 K16 K16_1MM*K16_1MM K16_1MM K14_1MM*K14_1MM K14	5030.68
18	K16*K16 K16 K16_1MM*K16_1MM K16_1MM	5034.28
19	K16*K16 K16 K10 K10*K10	5041.38
20	K16*K16 K16 K11*K11 K11	5039.68
21	K16*K16 K16 K12*K12 K12	5039.32
22	K16*K16 K16	5044.48
23	K16 K10	5100.4
24	K16 K11	5099.72
25	K16 K12	5101.34
26	K16	5103.6

Lisa C: Maskeerimisprogrammi käsureaparameetrid

-h/–help	trükib käsurea parameetrid
-l/–list	defineerib uue mudeli tunnuse (vt näited lisa D)
-lf/–lists_file	annab mudeli tunnused ette failina
-lp/–list_prefix	määrab mudelis kasutatavate k-meeride tabeli nimede eesliite
-p/–probability_cutoff	määrab maskeerimise nivoo ($[0, 1]$)
-a/–absolute_value_cutoff	määrab nivoo k-meeride sageduse põhiseks maskeerimiseks
-m5/–mask_5p	5'-suunas maskeeritavate nukleotiidide arv
-m3/–mask_3p	3'-suunas maskeeritavate nukleotiidide arv
-c/–masking_char	maskeerimiseks kasutatav tähemärk
-s/–soft_mask	lülitab sisse väiketähega maskeerimise
-d/–masking_direction	määrab maskeeritava ahela (fwd, rev, both)

Lisa D: Maskeerimisprogrammi käsurea näited

Näide 1. Mudeli tunnuste määramine käsurealt:
i. Mudel kasutab tunnust K6 koefitsiendiga 0,2 ja vabaliiget 4,0:
./masker -l homo_sapiens_6.list 0.2-l 4.0 < järjestus.fasta
ii. Mudel kasutab tunnust K12_1MM koefitsiendiga −2,5:
./masker -l homo_sapiens_12.list -2.5 1 < järjestus.fasta
iii. Mudel kasutab tunnust K16*K16 koefitsiendiga 1,0:
./masker -l homo_sapiens_16.list 1.0 0 sq < järjestus.fasta

Näide 2. Mudeli tunnuste määramine failiga:
./masker -lf mudel < järjestus.fasta

Faili *mudel* kuju (esimesel real on vabaliige):

```
4.0
homo_sapiens_11.list 0.4
homo_sapiens_16.list 0.5 1 sq
```

Näide 3. Vaikimisi mudeli kasutamine etteantud organismi k-meeride tabelitega
./masker -lp mus_musculus < järjestus.fasta

Näide 4. Maskeerida praimeri 3'-otsast 3 nukleotiidi 5'-suunas ja 1 nukleotiid 3'-suunas. Kasutada väiketähega maskeerimist.
./masker -lp mus_musculus -m5 3 -m3 1 -s < järjestus.fasta

Näide 5. Maskeerida ainult 5'→3'-suunaline ahel. Maskeerida tähemärgiga X.
./masker -lp mus_musculus -d fwd -c X < järjestus.fasta

Näide 6. Maskeerida kõik sellised positsioonid, millele vastavaid 12-meere on uuritavas genoomis rohkem kui 10.
./masker -l homo_sapiens_12.list -a 10 < järjestus.fasta

Lisa E: Primer3-e sisendi ja väljundi näited

Sisendjärjestused ei ole täispikkuses näidatud.

Sisend:

```
SEQUENCE_ID=test1
SEQUENCE_TEMPLATE=ATACATAATTAAATTATATATAATTAAATTATATAACATACTTATATACTAATTTAGCATACTTACAA...
PRIMER_NUM_RETURN=1
PRIMER_MASK_TEMPLATE=1
PRIMER_WT_FAILURE_RATE=1.0
PRIMER_MIN_THREE_PRIME_DISTANCE=1
=
```

Väljund:

```
SEQUENCE_ID=test1
SEQUENCE_TEMPLATE=ATACATAATTAAATTATATATAATTAAATTATATAACATACTTATATACTAATTTAGCATACTTACAA...
PRIMER_NUM_RETURN=10
PRIMER_MASK_TEMPLATE=1
PRIMER_WT_FAILURE_RATE=1.0
PRIMER_MIN_THREE_PRIME_DISTANCE=1
PRIMER_LEFT_NUM_RETURNED=1
PRIMER_RIGHT_NUM_RETURNED=1
PRIMER_INTERNAL_NUM_RETURNED=0
PRIMER_PAIR_NUM_RETURNED=1
PRIMER_PAIR_0_PENALTY=0.216014
PRIMER_LEFT_0_PENALTY=0.036952
PRIMER_RIGHT_0_PENALTY=0.179062
PRIMER_LEFT_0_SEQUENCE=AGGTCTCACTCATGTTGCCG
PRIMER_RIGHT_0_SEQUENCE=CTGGGTAGAGTGGCTCATGC
PRIMER_LEFT_0=655,20
PRIMER_RIGHT_0=763,20
PRIMER_LEFT_0_TM=60.037
PRIMER_RIGHT_0_TM=60.179
PRIMER_LEFT_0_GC_PERCENT=55.000
PRIMER_RIGHT_0_GC_PERCENT=60.000
PRIMER_LEFT_0_SELF_ANY_TH=0.00
PRIMER_RIGHT_0_SELF_ANY_TH=0.00
PRIMER_LEFT_0_SELF_END_TH=0.00
PRIMER_RIGHT_0_SELF_END_TH=0.00
PRIMER_LEFT_0_HAIRPIN_TH=0.00
PRIMER_RIGHT_0_HAIRPIN_TH=38.76
PRIMER_LEFT_0_END_STABILITY=5.6900
PRIMER_RIGHT_0_END_STABILITY=4.0600
PRIMER_PAIR_0_COMPL_ANY_TH=7.88
PRIMER_PAIR_0_COMPL_END_TH=0.00
PRIMER_PAIR_0_PRODUCT_SIZE=109
=
```

Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks

Mina, **Maarja Lepamets**,

1. annan Tartu Ülikoolile tasuta loa (lihtlitsentsi) enda loodud teose

PCR-i praimerite kvaliteedimudeli integreerimine Primer3-e disainimetoodikasse,

mille juhendaja on **Lauris Kaplinski**,

1.1. reprodutseerimiseks säilitamise ja üldsusele kättesaadavaks tegemise eesmärgil, sealhulgas digitaalarhiivi DSpace-is lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja lõppemiseni;
1.2. üldsusele kättesaadavaks tegemiseks Tartu Ülikooli veebikeskkonna kaudu, sealhulgas digitaalarhiivi DSpace´i kaudu kuni autoriõiguse kehtivuse tähtaja lõppemiseni.

2. olen teadlik, et punktis 1 nimetatud õigused jäävad alles ka autorile.

3. kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest tulenevaid õigusi.

Tartus, **05.11.2015**