Моделирање на UNO како марков процес

Вовед

Играта со карти уно поради нејзината стохастичка природа може да се моделира како вид на случаен процес. Во овој проект ќе се обидам да ја моделирам како вид на марков процес и да видам како должината на партиите во една игра зависи од различните фактори во истата.

За да го покажеме марковото својство прво потребно е да ги дефинираме карактеристиките на состојбата во секој момент. Секоја состојба ги има следните карактеристики: горната карта на шпилот за фрлање карти, картите во раката на играчот кој е на ред, шпилот од кој се влечат карти, насоката во која се движи редоследот на играње.

Доколку ги знаеме сите овие податоци тогаш можеме да ги одредиме сите идни можни состојби и соодветните веројатности. Постапката на следниот играч секогаш зависи потполно од постапката на тој кој е веќе на ред, но НЕ и од постапката на играчите пред него. Значи распределбата на идните потези зависи само од претходниот потег и ова го задоволува марковото својство.

Пример:

Најгорната карта моментално е 5 срце, во раката ние држиме 5 детелина, 9 срце и една карта за менување на боја, шпилот за влечење има уште 20 карти и насоката на играта е согласна со стрелките на часовникот. Веројатностите за следната состојба зависат само од овие информации. Не е битно дали 5 срце дошла од некој играч пред еден, два или 19 кругови наназад во играта и не е битно како и кога сме стигнале ние до картите кои ги држиме во моментот туку само дека тие се тие.

Чим го воспоставивме марковото својство, какви други карактеристики имаат состојбите?

Преодни состојби – поради огромното множество на можни состојби, веројатноста

една состојба да се повтори повеќе пати е занемарливо мала, што значи можеме да претпоставиме дека сите состојби во играта се преодни состојби, односно

Повратни состојби – поради тоа што состојбите се преодни, следува дека тие НЕ се повратни.

Апсорбирачки состојби – секоја состојба во која што системот останува во неа откога ќе влезе е апсорбирачка. Во овој случај, тоа е состојбата во која некој од играчите ќе победи. Кога еден играч ќе ги потроши сите свои карти системот останува во таа состојба.

Ергодичност – според тоа што постојат повеќе апсорбирачки состојби во системот, следува дека системот не е ергодичен, односно нема шанса системот да се најде во сите состојби барем еднаш.

Конечност – иако множеството на можни состојби е огромно, сепак тоа е конечно. Ова, комбинирано со постоењето на апсорбирачки состојби доведува до заклучок дека веројатноста веригата да е апсорбирана е 1, односно играта секогаш ќе заврши и ќе има победник.

Очекувано време на апсорбција – ова всушност е времето потребно да заврши партијата, односно некој да победи. Ова ќе го мериме во број на редови на играчите кои поминале се додека не се стигне до апсорбирачката состојба. Ова е всушност тоа на кое ќе се фокусираме низ текот на ова истражување.

Комуникација – речиси секоја една состојба е достижлива од секоја останата, па единствено ограничување тука би ни го претставиле апсорбирачките состојби. Сите тие си формираат своја одделена комуникациска класа во која може да се стига од една до друга состојба.

Карактеристики на уно како случаен процес

Понатаму, играта ќе ја разгледуваме како случаен процес од променливата t – број на играчи во играта. Конфигурирајќи различни правиле ќе се обидеме да го анализираме процесот преку зависноста на бројот на играчи.

Поради комплексноста на играта и обемот на множеството од состојби и врските помеѓу истите, постојат 2 опции за изучување.

1. Да се поедностави проблемот и играта, односно да се разгледа игра во која има 3 до 4 броеви на карти, 2 знаци и двајца играчи. Во ваков случај би можело да се генерира матрица за веројатност на премин помеѓу состојби и во идеален случај ригерозно да се најдат заклучоци за процесот.
2. Да се користат Монте Карло статистички методи за да се донесат заклучоци за вистинската игра или некоја и покомплицирана верзија од играта.

Во ова истражување ќе се задржеме на Монте Карло методот и ќе се обидеме да извлечеме заклучоци користејќи моделирање и генерирање случајни примероци од играта.

Моделот

За моделирање на играта ќе користиме главно python код. Реализирано е користејќи три различни класи, имено Card, Player и Game.

Card: Оваа класа користи за претставување на една карта, односно нејзиниот број и нејзиниот знак.

Player: Оваа класа користи за претсатвување на играч и сите негови особини. Еден играч е претставен со колекција од карти кои ги држи во рака.

Game: Ова е главната класа во која е вградена логиката на играта. Една инстанца се карактеризира со голем број на подесувања, најбитни од кои се бројот на играчи (2, 3, 4, ...), нивото на мешање на картите (low, medium, high), бројот на броеви на карти (стандардно 14), бројот на знакови (стандардно 4) и дали се користат специјални карти.

Правилата на играта се следните:

* На почетокот се меша шпилот и се вади една карта на средина.
* Се делат карти на секој од играчите чии ред се менува кружно.
* Кога ќе дојде на ред еден играч, може да фрли карта доколку нејзиниот знак или нејзниниот број се совпаѓа со картата која се наоѓа најгоре на шпилот од фрлени карти. Доколку не поседува таква карта, влечи 1 карта и неговиот ред поминува.
* Специјални карти се 1 и 8 (скокање), 7 и 14 (+2 и +4 на следниот играч), 10 (менување на редоследот на кругот).
* Играта прекинува кога 1 играч ќе ги фрли сите карти. Тогаш тој победува.

Целиот код со кој се реализа ова се наоѓа на следниот линк:

<https://github.com/borjannn/uno_stochastic_process>

Симулации

Симулациите се главно поделени на две групи:

* Игри во кои се користат специјални карти (скокање, полнење и менување редослед)
* Игри во кои НЕ се користат специјални карти

Поради ограничениот борј на карти (4\*14) во првичните симулации ќе ја разгледаме должината на партиите само за броеви на играчи 2 до 9 бидејќи со почетен број на карти 6 максимум 9 луѓе можат да играат и при тоа да останат карти во шпилот за влечење.

Главни својства кои ќе ги набљудуваме ќе се математичкото очекување на должината на партијата и стандардната девијација на истата. За таа цел ќе користиме точкасти оценувачи за својствата. За математичкото очекување ќе го користиме просекот, а за стандардната девијација коренот од средната квадратна разлика со именител n-1 за пристрасност.

Следните експерименти се изведени врз по 20 000 примероци од секој број на играчи од 2 до 9.

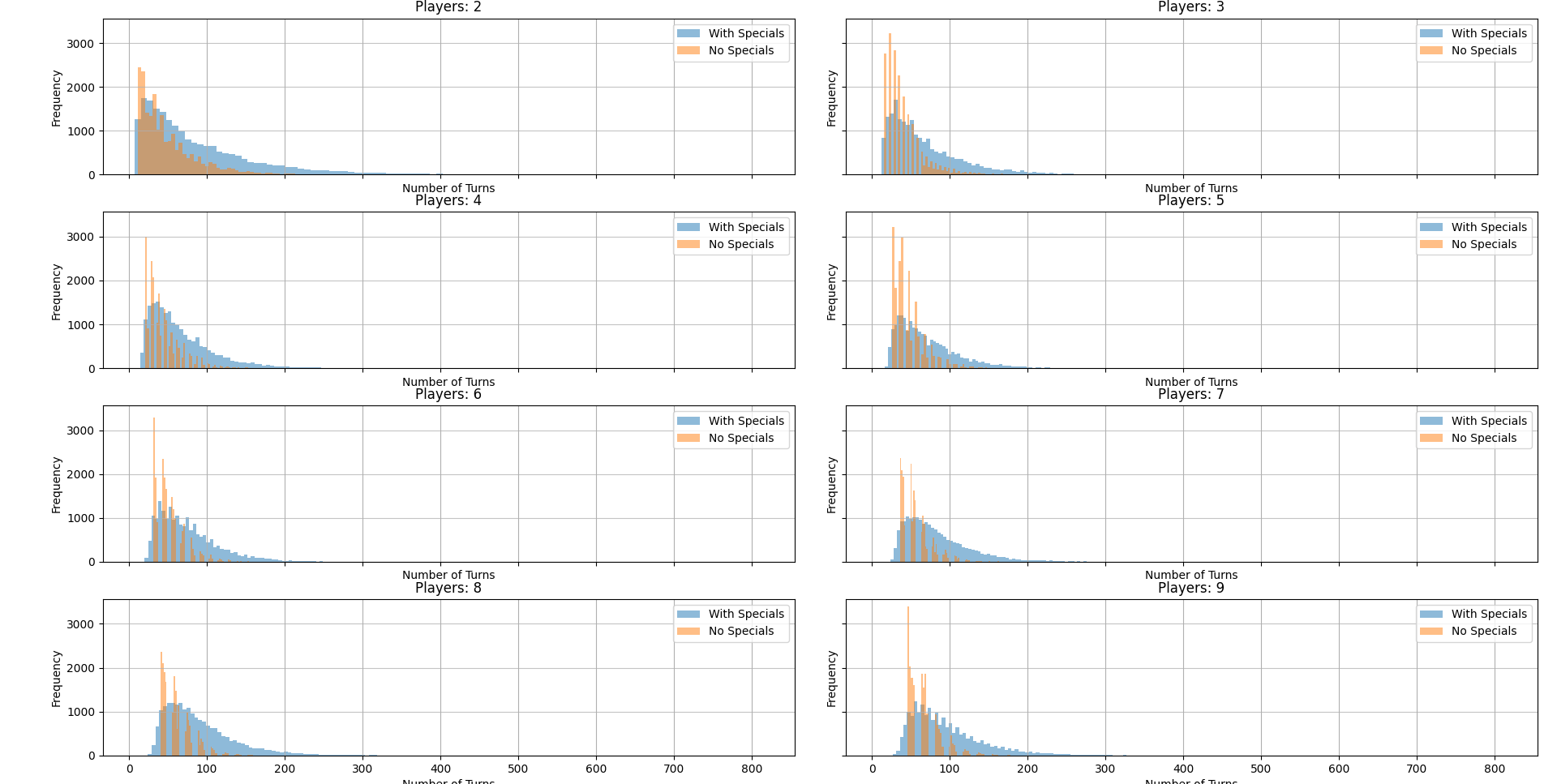
Хипотеза

За сега не е можно да предвидеме како ќе личат овие графици, но можеме да направиме препоставка дека дисперзијата на случајната променлива ќе е поголема кај игрите во кои се игра со специјални карти. Оваа претпоставка ја правам врз основа на тоа што специјалните карти ќе додадат повеќе хаотичност и случајност во системот, па должините на партиите повеќе ќе варираат.

Густината на случајната променлива

Нека случајната променлива ја дефинираме како должината на партијата ако играат играчи. Тогаш ни е случаен процес дефиниран над множеството . За да ја процениме густината на случајната променлива ги поделиме вредностите добиени од симулациите на помали интервали (опсези) и ќе ги групираме во истите со цел полесно да ги претставиме. На x оската ќе ги имаме интервалите а на у, бројот на примероци од случајно генерираните кои припаѓаат во тие интервали.

Резултати



Забелешка 1:

Хипотезата е потврдена, односно податоците кои доаѓаат од игрите во кои се дозволени специјални карти се многу повеќе расеани, а од друга страна тие каде не се дозволени се поконзистентни и збиени.

Забелешка 2:

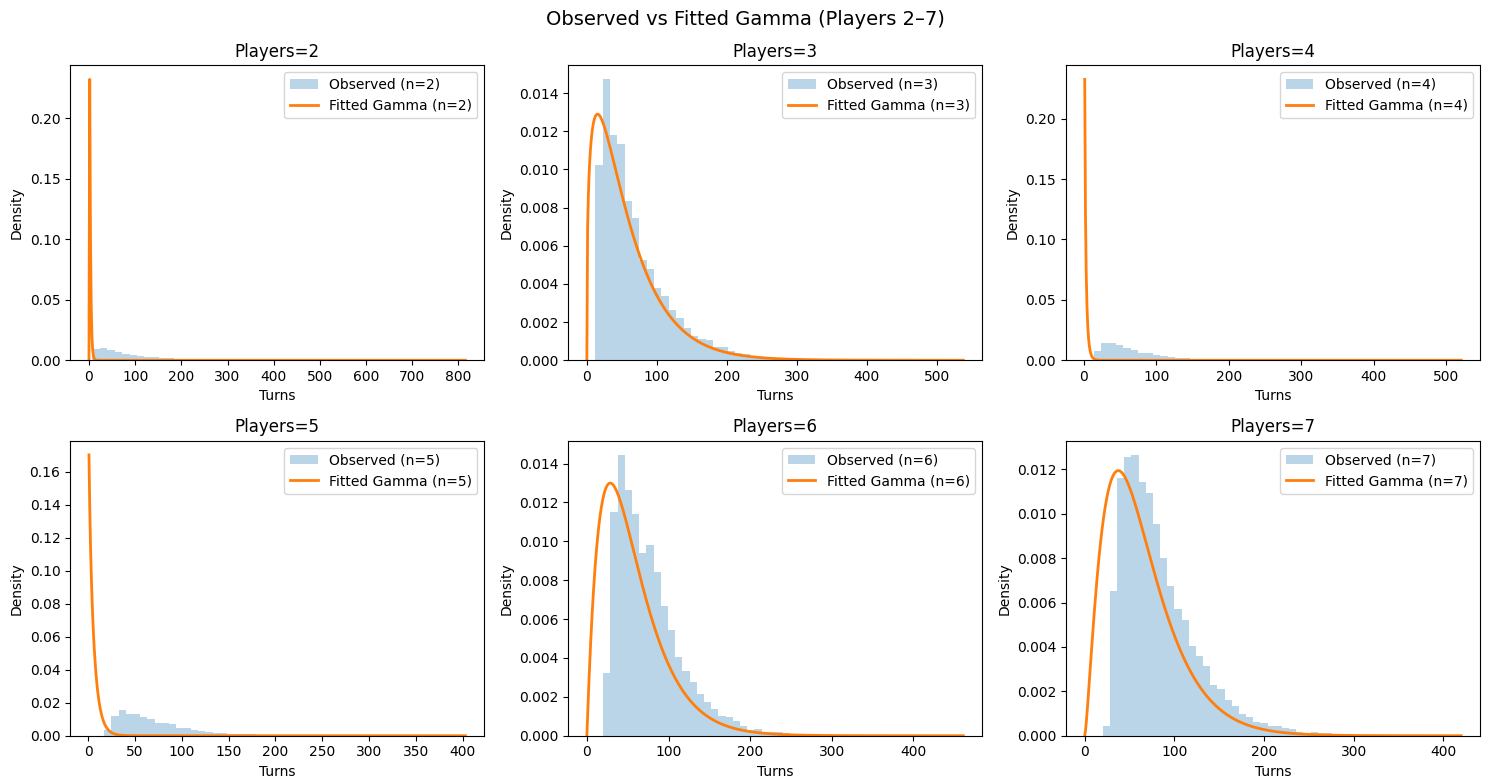
Како што се зголемува бројот на играчи,помалку варираат податоците и повеќе се стабилизираат. Распределбата личи како да се приближува кон кормална распределба на ист начин како што личи хиквадрат распределбата кога растат нејзините степени на слобода. Според ова, можеби смееме да направиме хипотеза дека распределбата на оваа случајна променлива е некој вид на гама распределба.

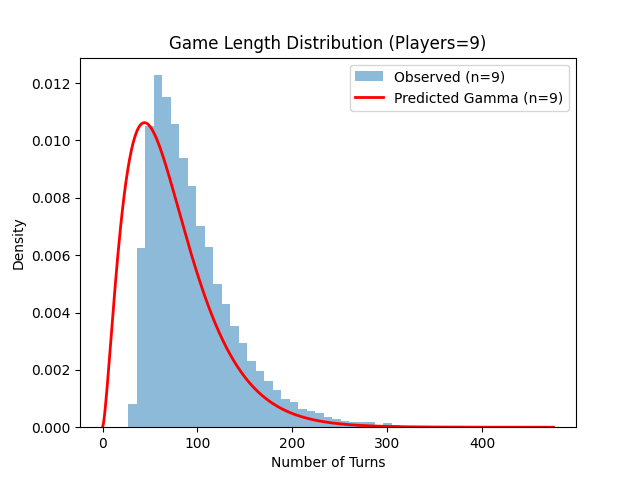
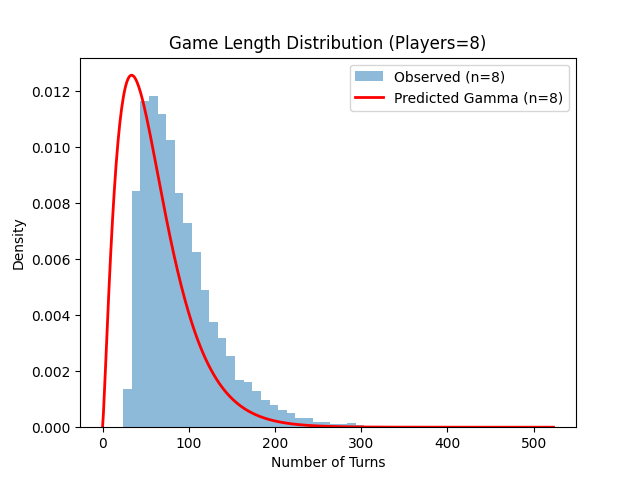
За жал, поради ограниченоста со бројот на играчи, оваа хипотеза не можеме со сигурност да ја потврдиме, но можеме да се обидеме преку некои предвидувања за параметрите.

Споредба со Гама распределба

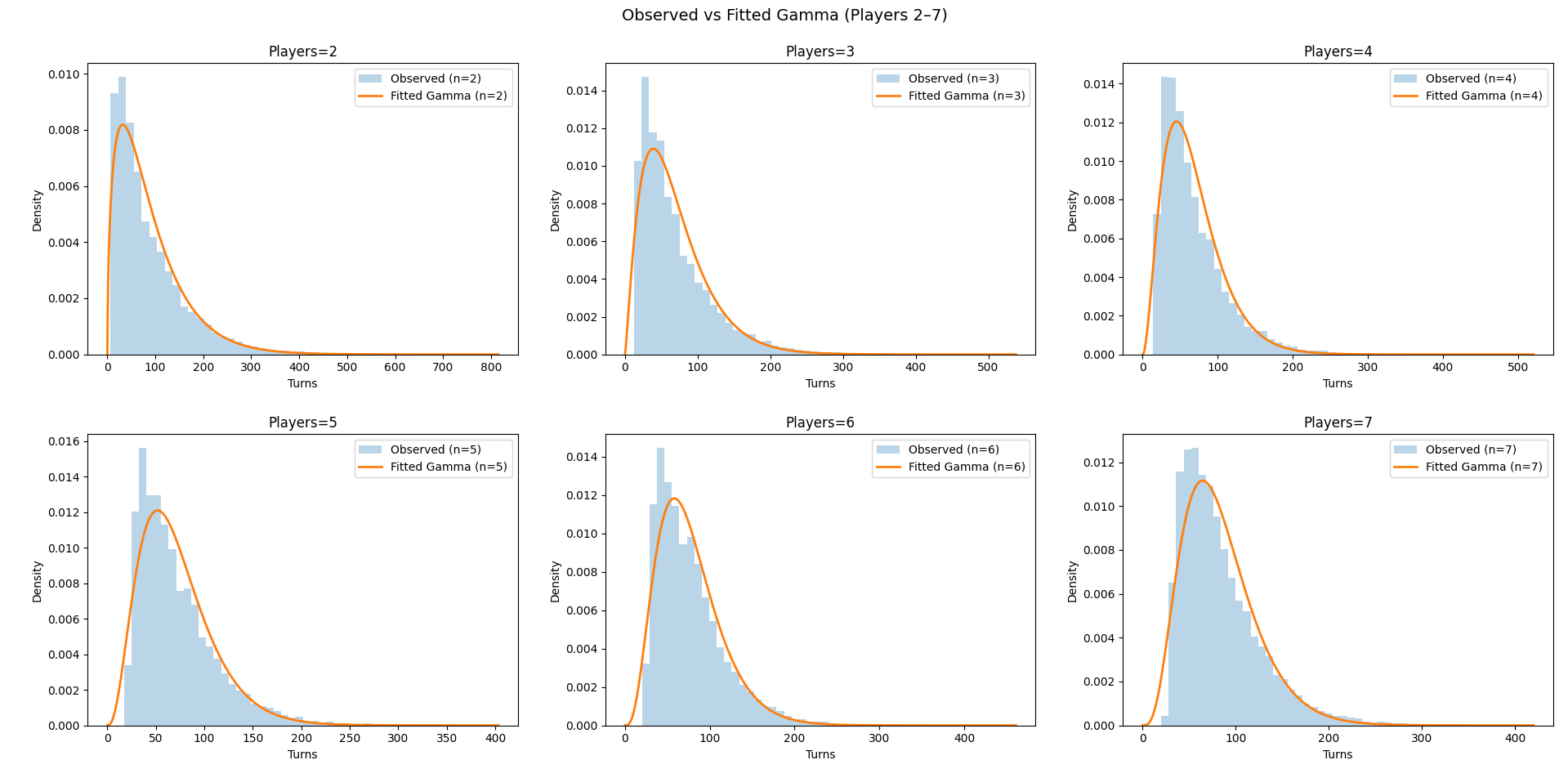
Сега ќе се обидеме да ги искористиме податоците од распределбите на 2, 3, ... , 7 играчи за да го предвидеме изгледот на распределбата за 8 и 9 играчи. За оваа цел, прво потребно е да искористиме оценувачи за гама распределбата и да ги најдеме нејзините параметри користејќи ги податоците, и потоа користејќи некој статистички метод да ги најдеме тие за 8 и за 9.

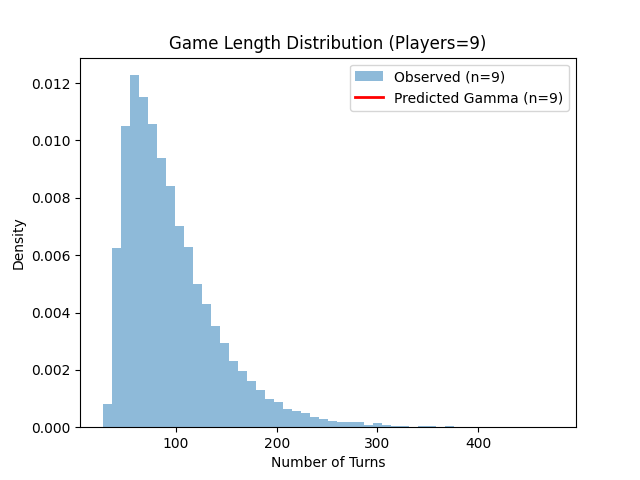
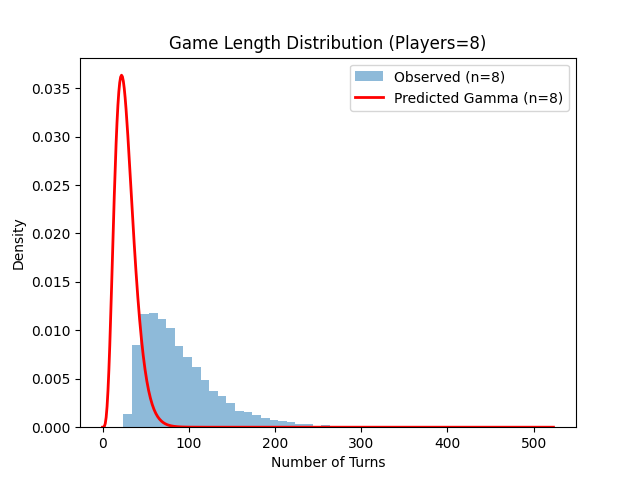
Резултати





Иако предвидените вредности личат добри, сепак имаме проблем бидејќи во оценувањето на распределбите на број на играчи 2, 4 и 5 налетуваме на многу чест проблем кај максимално подобниот оценувач за гама распределбата. Ова можеме да го поправиме со предефинирање на loc параметарот да е строго позитивен и ги добиваме следните резултати:





Како што гледаме, во општо не се поклопуваат за 8, а за 9 играчи пак, распределбата во општо не постои будејќи најверојатно регресијата нашла опаѓачки тренд и тоа довело до негативен параметар.

Ова не мора конкретно да значи дека процесот не е во природа вид на Гама распределба, туку може само да значи дека врската помеѓу параметрите и бројот на гирачи не е полиномна.

Математичко очекување и дисперзија

Следно, ќе испитаме како се однесуваат математичкото очекување и дисперзијата на променливата во зависност на бројот на играчи, дали се користат специјални карти, и степенот на мешање на шпилот.

Хипотеза

Дисперзијата на променливата, како и нејзиното математичко очекување ќе се зголемува како што подобро го мешаме шпилот карти. Подоброто мешање ќе доведе до повеќе неред, и резултантно помалце предвидливост во играта, па повеќе варијација во должината на партиите. Поради истите причини, истото би требало да се случи и како што го зголемуваме бројот на играчи.

За да ја испитаме оваа хипотеза, ќе ги претставиме на график дисперзијата, и математичкото очекување на податоците на у оска, а на х оска бројот на играчи. Ќе претставиме податоци од сите комбинации меѓу (low shuffle, medium shuffle , high shuffle) и (special, no special), односно 6 класи.

