**Dave Ashley’s Notes on Blackjack**

***and***

**Documentation of the *BLACKJACKSIM* Program**

David T. Ashley (DASHLEY@GMAIL.COM)

Version 0.1a, October 27, 2010

Table of Contents

[1 Introduction and Overview 4](#_Toc277704114)

[2 Overview of Blackjack 5](#_Toc277704115)

[3 Overview of Card Counting 6](#_Toc277704116)

[3.1 General Description 6](#_Toc277704117)

[3.2 Human Complexity Limits 6](#_Toc277704118)

[*4* *BLACKJACKSIM* 8](#_Toc277704119)

[4.1 Platform Requirements 8](#_Toc277704120)

[4.2 Compilation 8](#_Toc277704121)

[4.3 Running *BLACKJACKSIM* 8](#_Toc277704122)

[4.4 Command-Line Arguments 8](#_Toc277704123)

[4.4.1 Command-Line Options Specifying General Behavior 9](#_Toc277704124)

[4.4.2 Command-Line Options Specifying the Variant of Blackjack to be Analyzed 9](#_Toc277704125)

[4.4.3 Command-Line Options Specifying Major Actions 12](#_Toc277704126)

[4.4.4 Command-Line Options Specifying Major Action Behavior 12](#_Toc277704127)

[4.5 Description of Output 12](#_Toc277704128)

[5 Strategy for H17, 6 Decks, 75% Penetration 12](#_Toc277704129)

[5.1 Correct Basic Strategy 12](#_Toc277704130)

[5.2 Hi-Lo Card Counting System Analysis 12](#_Toc277704131)

[5.2.1 Expected Value of a Hand as a Function of the True Count 12](#_Toc277704132)

[5.2.2 Expected Value of a Hand as a Function of the True Count and Shoe Penetration 12](#_Toc277704133)

[5.2.3 Indexed Playing Strategy 13](#_Toc277704134)

[5.2.4 Insurance Betting Strategy 13](#_Toc277704135)

[5.2.5 Penalty for Using the Play True Count for Insurance Betting 13](#_Toc277704136)

[5.2.6 Strategy for Unobserved Burn Cards 13](#_Toc277704137)

[5.2.7 Advantage of Using Odd Betting Amounts 13](#_Toc277704138)

[5.2.8 Shoe Abandonment Criteria 13](#_Toc277704139)

[5.2.9 Timid Betting Effectiveness 13](#_Toc277704140)

[5.2.10 Aggressive Betting Effectiveness 13](#_Toc277704141)

[5.2.11 Superaggressive Betting Effectiveness 13](#_Toc277704142)

[5.3 Most Effective Card Counting System Analysis 13](#_Toc277704143)

[5.3.1 Expected Value of a Hand as a Function of the True Count 13](#_Toc277704144)

[5.3.2 Expected Value of a Hand as a Function of the True Count and Shoe Penetration 13](#_Toc277704145)

[5.3.3 Indexed Playing Strategy 13](#_Toc277704146)

[5.3.4 Insurance Betting Strategy 13](#_Toc277704147)

[5.3.5 Penalty for Using the Play True Count for Insurance Betting 13](#_Toc277704148)

[5.3.6 Strategy for Unobserved Burn Cards 13](#_Toc277704149)

[5.3.7 Advantage of Using Odd Betting Amounts 14](#_Toc277704150)

[5.3.8 Shoe Abandonment Criteria 14](#_Toc277704151)

[5.3.9 Timid Betting Effectiveness 14](#_Toc277704152)

[5.3.10 Aggressive Betting Effectiveness 14](#_Toc277704153)

[5.3.11 Superaggressive Betting Effectiveness 14](#_Toc277704154)

[6 Miscellaneous Topics 14](#_Toc277704155)

[6.1 Avoiding Casino “Heat” 14](#_Toc277704156)

[7 Index 15](#_Toc277704157)

# Introduction and Overview

This document provides my personal notes on the game of blackjack and documentation for the *BLACKJACKSIM* program.

*BLACKJACKSIM* is a program that evaluates the effectiveness of various blackjack playing and betting strategies. *BLACKJACKSIM* is designed to run only on *\*nix[[1]](#footnote-1)* platforms, and may in some cases use many hours of CPU time to produce results.

The full source code of the *BLACKJACKSIM* program is provided at the same website as this document under the GNU General Public License.

This document is not a blackjack tutorial or a mathematical tutorial. It is assumed that the reader is *very* familiar with blackjack, and understands concepts of algebra, calculus, statistics, and linear algebra.

In 2010, I became an avid blackjack player, and later learned and attempted to play using the hi-lo method of card counting.

During this period, I was exposed to many different theories from other blackjack players.

I had poor results card counting during this period, and found that my results didn’t seem to be any better card counting than when using a simple basic strategy card. However, I did consider the possibility that my poor results were partly due to these factors:

* Not knowing the precise true card count at which the advantage shifts from the house to the player.
* Not being familiar with the correct indexed play strategies.
* Betting rather timidly, even with a very positive card count. (At a $10 table, I would rarely bet above $25 per hand.)

I do not trust the information found on blackjack websites or in books. It seems plausible that most of the site authors are poor mathematicians and have copied the information they have from other websites and from books. It would be possible for incorrect information to appear on many websites by that mechanism.

Based on my exposure to other players, my personal experiences, and my distrust of information found on blackjack websites and in books, I decided to write a simple program (*BLACKJACKSIM*) to answer the following questions:

* Are published basic strategy charts accurate? (Some players claim that basic strategy charts are produced by the casinos and are deliberately inaccurate to enable the casinos to make more money.)
* What is the standard deviation (or some equivalently telling measure) when playing using basic strategy?
* Are the published house advantages for the published basic strategy charts accurate?
* Can the variant of blackjack that I play actually be won, and if so, how? (My initial negative experiences with card counting gave me doubts.)
* Which is the most effective card counting strategy for the variant of blackjack I typically play? (Learning a card counting strategy is a substantial investment—it makes sense to learn only the best one.)
* What is the best indexed playing strategy? (This information seems to be scarce on the internet. Additionally, the information on the internet seems to be only in terms of integer values of the true count. Players can often calculate the true count to greater accuracy, so more precise information would potentially be helpful.)
* What is the most effective betting strategy, and what is the tradeoff between rate of return and standard deviation?
* Does the accuracy of card counting vary with how far into the shoe the count is being used?
* How much do unobserved burn cards affect the player’s advantage? (At the local casino where I play, when dealer changes occur during a shoe, a card is burned and this is not shown to the players.)
* What is the best strategy for adjusting the card count when an unobserved card is burned?
* Are there any other strategies that can be used to increase the player’s advantage? Such strategies might include:
	+ Ignoring dealer mistakes that are favorable to the player.
	+ Using $2.50 chips. (At the local casino I visit, $2.50 chips are paid $4 on a blackjack—a $0.25 overpayment because the casino does not have the chip denominations to pay the player $3.75. Similarly, a bet of $102.50 would be paid $154 on a blackjack. Generally, bets involving $2.50 chips are annoying to the dealer, but the question of how much advantage these confer is worthy of investigation.)

The variant of blackjack I am most interested in is the variant played at the local casino I visit. However, the BLACKJACKSIM program can be used to investigate the properties and optimal playing strategies for other variants of blackjack.

# Overview of Blackjack

An overview of blackjack isn’t provided because anyone reading this document presumably already knows the rules of blackjack.

Before reading further, you should be familiar with the following aspects of blackjack:

* The basic rules of play.
* The meaning of the terms *card*, *face card*, *ace*, *deck*, *shoe*, *hand*, *stand*, *hit*, *double down*, *split*, *insurance*, *even money*, *surrender*, and *basic strategy*.
* The distinction between *soft totals* and *hard totals*.
* The distinction between a *natural* (or natural blackjack) and cards that total twenty-one but are not a natural blackjack.

If the terms above are not familiar, *Wikipedia* has an excellent short entry for blackjack, and a search of the Internet by the terms above should produce useful information. Alternatively, a visit to the local casino and a few questions posed to a dealer or pit boss should impart familiarity with the topics above.

# Overview of Card Counting

## General Description

*Card counting* in blackjack is the process of observing all cards that are dealt from a shoe and using this information to make inferences about the statistical character of those cards that remain in the shoe.

Card counting is effective only in games that are dealt from a shoe. Card counting is ineffective in games dealt from a continuous shuffling machine, as cards are immediately placed back into the continuous shuffling machine.

## Human Complexity Limits

A computer observing the cards dealt from a shoe could track how many of each value of card are left in the shoe, and make betting and playing decisions based on the precise composition of the cards remaining in the shoe. However, tracking dealt cards this precisely and making complex betting and playing decisions based on this information is unattainable for most humans.

Card counting systems, out of necessity, have to be far simpler than tracking the precise composition of the remaining cards in the shoe. Card counting systems are well correlated with ideal play, but they are not ideal.

The *BLACKJACKSIM* program is designed to evaluate and tune only card counting systems with these characteristics:

* The system involves a maximum of two counts, both rational numbers, one used for betting and one used for play. The count used for betting is denoted cb and the count used for play is denoted cp.
* cb is assumed to be approximately monotonic with respect to the expected value of the hand. cp need not have this property.
* Based on the assumed monotonicity, a threshold can be calculated, cbth, at or above which the expected value of the hand is greater than 1.
* cb, cp , and cbth are rounded to a certain granularity (typically 1/10 but easily configurable in the program source code for each counting system) before being used. This simulates the limited ability of a human player to make calculations while playing blackjack, and also provides a framework for the creation of bins for tabulating the results of the simulation as a function of cb and cp.
* The wager amount can be calculated as a function of cb in this way:

 
* cb and cp are arbitrary functions of the cards that have been observed dealt from the shoe. (In practice, both are linear combinations of integer values assigned to each card, perhaps adjusted by dividing by the number of decks remaining in the shoe.)
* cb and cp may start at an arbitrary offset (not necessarily zero) at the start of the shoe.

Insurance and Even Money

Even money is separate from insurance, as with even money there is one less ten in the deck.

## Betting Strategy

It makes no sense to bet *any* amount of money when the expected value of the hand is 1 or less. However, casinos generally will not allow a player to arbitrarily sit out hands. It is typically necessary to bet *some* amount of money on every hand. The exception is that breaks to visit the restroom or obtain another drink are tolerated.

When the expected value of the hand is 1 or less, it makes sense to bet only the table minimum. However, when the expected value of the hand is greater than 1, the amount to bet should be increased. The amount to increase the bet is a tradeoff between variability and rate of return. The more aggressive the increases, the higher the rate of return, but the larger the variability of the profit or loss during an individual gambling session.

The most aggressive strategy possible is to use a step-jump from table minimum to the table maximum when the true card count reaches a threshold implying that the expected value of a hand is greater than 1. This strategy is depicted in Figure 1.



Figure : Maximally Aggressive Bipolar Betting Strategy

At a typical casino, the betting strategy depicted in Figure 1 would attract attention. A typical table minimum is $10, and a typical table maximum is $500. Alternating between those two extremes would attract substantial attention.

A second less aggressive bipolar strategy is shown in Figure 2. When cb≥cbth, instead of the table maximum, a lesser amount is bet. At a table with a $10 minimum, the bet when cb≥cbth might be $25 or $100. Alternating bets between $10 and $25 or between $10 and $50 will not usually attract attention. Alternating bets between $10 and $100 may attract attention.



Figure : Less Aggressive Bipolar Betting Strategy

## Indexed Play

# *BLACKJACKSIM*

## Platform Requirements

## Compilation

## Running *BLACKJACKSIM*

*BLACKJACKSIM* may be executed using the command-line “*blackjacksim <options>*”, where the options are as specified in §4.4.

*BLACKJACKSIM* produces multiple output files, which are placed in the current working directory or in the location specified by the *-od* option.

To avoid naming conflicts between files, the program suffixes every file name with a timestamp of the form *yyyymmdd\_hhmmss*, indicating the invocation time of the program. This behavior makes it possible to run the program many times sequentially using the same output directory.[[2]](#footnote-2)

*BLACKJACKSIM* will produce a base file designed to be viewed with a web browser, and this base file will link to other graphics files produced. The base file is named *blackjacksim\_output\_yyyymmdd\_hhmmss.html*.

Other than the heavy CPU consumption that will result, there is no barrier to running multiple instances of the program at the same time on the same machine.

## Command-Line Arguments

BLACKJACKSIM command-line arguments are of four types:

* Those that specify the general behavior of the program.
* Those that specify the variant of blackjack to be analyzed.
* Those that that specify the major actions to be performed.
* Those that specify how a major action is to be performed.

Command-line arguments may be specified in any order. Conflicting or duplicate command-line arguments will result in an error message and program termination.

Most command-line arguments have default values, so that command-line arguments only need to be specified if the default behavior is to be modified.

### Command-Line Options Specifying General Behavior

**-od *path***

Use path as the directory in which to create temporary files and output files.

**-v *n***

Specifies the verbosity level, where 0 ≤ n ≤ 9. 0 is the least verbose, and 9 is the most verbose. The default is verbosity 0.

**-debugfile**

Indicates that *BLACKJACKSIM* should produce a debugging file rather than output all information specified by the *–v* option to the console. The file will be named *blackjacksim\_debug\_yyyymmdd\_hhmmss.txt*. The default is not to produce a debugging file.

**-prng *prng\_spec***

Specifies the pseudo-random number generator to be used for all aspects of the program that require random numbers. The available choices are:

* *urandom* (default)
The /dev/urandom device should be used. This is a practical random number source, as it is efficient and the calls to read from this device are non-blocking.
* *random*
The */dev/random* device should be used. The calls to read from this device are blocking, and this choice will result in impractically slow program execution.

In the future, *BLACKJACKSIM* may be expanded to include other pseudo-random number generators if any shortcomings are discovered in */dev/urandom* or if the program is modified to compile for Windows systems.

### Command-Line Options Specifying the Variant of Blackjack to be Analyzed

#### House Behavior

**-dhs17** (default)

**-dss17**

*-dhs17* indicates that a dealer hits soft 17. *-dss17* indicates that the dealer stands on soft 17.

**-shoedecks *n*** (default: 6 decks)

Specifies the number of decks of cards in the shoe, 0.25 ≤ n ≤ 100. The default value is 6. The number of decks may be specified with a fractional portion of .25, .5, or .75. Each fractional increment of 0.25 will cause all cards of one suit to be present in the shoe.

This option is provided for more insightful graphs of effects that are a function of the number of decks in the shoe.

**-shoepennormal *n1 n2*** (default: *n1*/*n2* = 221/260 cards = 4.25/5 deck penetration)

**-shoepenuniform *n1 n2***

Specifies the number of cards that will be dealt before the dealer reshuffles after the end of the hand in play.

*-shoepennormal n1 n2* specifies a normal (i.e. bell-shaped) distribution such that 95% of the placements of the reshuffle card fall between *n1* and *n2* cards. This corresponds to typical casino behavior where dealers place the reshuffle point imprecisely.

*-shoepenuniform n1 n2* specifies a uniform distribution of the placement of the reshuffle card between *n1* and *n2* cards.

With either distribution, if *n1*=*n2*, the reshuffle point will be placed precisely at that number.

It is required that *n1* and *n2* are both at least 1, and that *n1* and *n2* are no greater than 52 times the number of decks specified via the *-shoedecks* option.

If the simulation runs out of cards from the shoe (which may happen if the deck penetration is specified as a large fraction), additional shuffled decks are used so that the hand can be completed.

**-burncardfirst** (default)

**-burncardfirstno**

Specifies whether or not the first card of the shoe is “burned”.

**-burncardfirstview** (default)

**-burncardfirstnoview**

Specifies whether the players are allowed to view the initial card in the shoe that is traditionally burned. At the local casino where I play, players are allowed to view this card.

**-burncardmiddleuniformdist *n1* *n2*** (default: *n1*/*n2*=520/624, i.e. a card is burned very 10 to 12 decks)

This specifies how often a non-first card from the shoe is “burned”. The distribution that can be specified is uniform in [*n1*, *n2*]. If *n1*=*n2*, a non-first card will be burned every *n1*=*n2* cards. It is typical that when a dealer is relieved for break or comes back from break, the new dealer “burns” the next card in the shoe. How often this happens varies.

If *n1*=*n2*=0, no “middle” cards are burned.

**-burncardmiddleview**

**-burncardmiddlenoview** (default)

Specifies whether the players are allowed to view the non-initial cards in the shoe that are traditionally burned when the dealer changes. At the local casino where I play, players are not allowed to view these cards without special permission from the pit boss.

**-blackjackpayout n1 n2** (default: *n1*=3, *n2*=2)

The blackjack payout, expressed as the rational number *n1*/*n2*. The traditional payout for blackjack is 3:2, but some casinos use 6:5 (which is less favorable to the player). 1 ≤ *n1* ≤ 1000 and 1 ≤ *n2* ≤ 1000.

**-insurancenone**

**-insuranceplayernaturalonly**

**-insuranceanyhand** (default)

Specifies whether insurance is not available, whether insurance is only offered if the player has a natural, or whether insurance is available on any hand that the player holds.

**-splitsmax** (default: 3)

Specifies the maximum number of times that a pair of like-valued cards may be split. Note that 3 splits will result in playing 4 hands simultaneously.

**-doubledown91011**

**-doubledown1011**

**-doubledownany** (default)

Specifies whether a player may double down only when his card total is 9, 10, or 11; may double down only when his total is 10 or 11; or may double down on any total.

**-doubleaftersplit** (default)

**-doubleaftersplitno**

Specifies whether a player may double down after splitting.

**-surrender**

**-surrenderno** (default)

*-surrender* specifies that the player may surrender his first two cards if the dealer does not have a natural in exchange for retaining half his bet. *-surrenderno* specifies that late surrender is not offered.

**-betmin *n*** (default: 10)

**-betmax *n*** (default: 500)

The minimum and maximum bets allowed at the table. The units are presumed to be dollars.

#### Environment

**-nplayers *n*** (default: 4)

Specifies the number of players at the table (1 ≤ n ≤ 50), including the simulated card counter and not including the dealer.

**-cardcounterposn *n*** (default: 3)

The position of the simulated card counter. A lower integer specifies that the player receives his cards earlier, i.e. is further to the right at the table as viewed from the players’ side of the table. N must be at least 1, but no larger than specified for *-nplayers*.

#### Player Behavior

**-betnoadvantageamount** (default: 10)

**-betadvantagebase** (default: 25)

**-betadvantageslope *n1 n2*** (default: 1)

*-betdisadvantageamount* specifies the amount that the simulated player bets when there is no advantage for the player. Typically, this is the table minimum.

*-betadvantagebase* specifies the base amount that the player bets when there is any advantage for the player.

*-betadvantageslope* specifies how rapidly the wager amount increases with respect to the card count, as a rational number. n1/n2=1 specifies that an increase in the card count of 1 will result in a wager increase of *-betadvantagebase*.

### Command-Line Options Specifying Major Actions

**-countingsystemnone**

**-countingsystemhilo** (default)

**-countingsystemomegaii**

The card counting system to be used for the simulation and evaluation. *-countingsystemnone* evaluates the best basic strategy for the game parameters.

### Command-Line Options Specifying Major Action Behavior

**-checkstrategy**

Checks the strategy developed by modifying each entry and evaluating system performance. This consumes much CPU time.

## Description of Output

# Strategy for H17, 6 Decks, 75% Penetration

## Correct Basic Strategy

## Hi-Lo Card Counting System Analysis

### Expected Value of a Hand as a Function of the True Count

### Expected Value of a Hand as a Function of the True Count and Shoe Penetration

### Indexed Playing Strategy

### Insurance Betting Strategy

### Penalty for Using the Play True Count for Insurance Betting

### Strategy for Unobserved Burn Cards

### Advantage of Using Odd Betting Amounts

### Shoe Abandonment Criteria

### Timid Betting Effectiveness

### Aggressive Betting Effectiveness

### Superaggressive Betting Effectiveness

## Most Effective Card Counting System Analysis

### Expected Value of a Hand as a Function of the True Count

### Expected Value of a Hand as a Function of the True Count and Shoe Penetration

### Indexed Playing Strategy

### Insurance Betting Strategy

### Penalty for Using the Play True Count for Insurance Betting

### Strategy for Unobserved Burn Cards

### Advantage of Using Odd Betting Amounts

### Shoe Abandonment Criteria

### Timid Betting Effectiveness

### Aggressive Betting Effectiveness

### Superaggressive Betting Effectiveness

# Miscellaneous Topics

## Avoiding Casino “Heat”

# Index

GNU General Public License 3

1. *\*nix* is the way that Unix and Unix-like systems are denoted. The *BLACKJACKSIM* program was developed and run on a lightly loaded Linux server; but it should compile and run on any Unix or Unix-like platform. [↑](#footnote-ref-1)
2. *BLACKJACKSIM* does not cover the unusual case of multiple instances of the program attempting to use the same timestamp as a basis for file naming, as might occur if two instances of the program are run approximately simultaneously with the same output directory. In this case, the first instance of the program that determines that a file it wishes to create already exists will terminate with an error message. [↑](#footnote-ref-2)