```
set.seed(131131)
wald_co_prob <- function(n, p, alpha) {</pre>
    z \leftarrow qnorm(1 - alpha / 2)
    k \leftarrow 0:n
    p_hat <- k / n
    condition <- sqrt(n) * abs(p_hat - p) / sqrt(p_hat * (1 - p_hat)) <= z</pre>
    # Sumowanie iteracji
    co <- sum(dbinom(k, n, p) * condition)</pre>
    return(co)
}
p < -0.5
alpha <- 0.05
n_values <- 10:100
co_probs <- sapply(n_values, function(n) wald_co_prob(n, p, alpha))</pre>
result <- data.frame(n = n_values, p_n = round(co_probs, 3))
print(result)
##
            p_n
       n
## 1
      10 0.891
## 2
      11 0.935
## 3
      12 0.854
      13 0.908
## 4
## 5
      14 0.943
## 6
      15 0.882
## 7
      16 0.923
## 8
      17 0.951
## 9
      18 0.904
## 10 19 0.936
## 11 20 0.959
## 12 21 0.922
## 13 22 0.948
## 14 23 0.907
## 15 24 0.936
## 16 25 0.957
## 17 26 0.924
## 18 27 0.948
## 19 28 0.913
## 20 29 0.939
## 21 30 0.957
## 22 31 0.929
## 23 32 0.950
## 24 33 0.920
## 25 34 0.942
## 26 35 0.959
## 27 36 0.935
## 28 37 0.953
## 29 38 0.927
## 30 39 0.947
## 31 40 0.919
## 32 41 0.940
```

```
## 33 42 0.956
## 34
      43 0.934
## 35
       44 0.951
## 36
       45 0.928
## 37
       46 0.946
## 38
       47 0.921
## 39
       48 0.941
## 40
       49 0.956
## 41
       50 0.935
## 42
       51 0.951
## 43
       52 0.930
       53 0.947
## 44
## 45
       54 0.924
## 46
       55 0.942
## 47
       56 0.956
## 48
       57 0.937
## 49
       58 0.952
       59 0.933
## 50
## 51
       60 0.948
## 52
       61 0.928
## 53
       62 0.944
## 54
       63 0.957
       64 0.940
## 55
## 56
       65 0.954
## 57
       66 0.936
## 58
       67 0.950
## 59
       68 0.932
## 60
       69 0.947
## 61
       70 0.928
## 62
       71 0.943
## 63
       72 0.956
       73 0.940
## 64
## 65
       74 0.953
## 66
       75 0.936
       76 0.950
## 67
## 68
       77 0.932
## 69
       78 0.946
## 70
       79 0.958
## 71
       80 0.943
       81 0.955
## 72
## 73
       82 0.940
## 74
       83 0.952
```

75

76

77

78

79

80

81 ## 82

83

84

85

84 0.937

85 0.950

86 0.934

87 0.947

88 0.931

89 0.94490 0.955

91 0.941

92 0.953

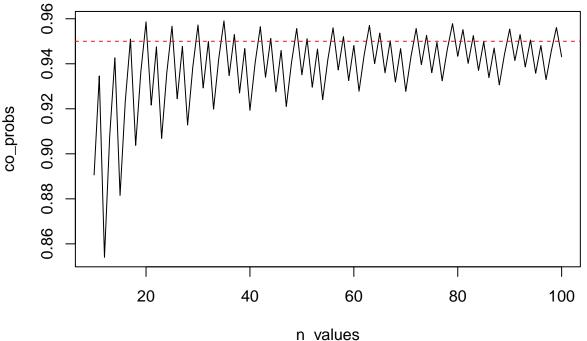
93 0.939

94 0.951

86 95 0.936

```
## 87 96 0.948
## 88 97 0.933
## 89 98 0.946
## 90 99 0.956
## 91 100 0.943

plot(n_values, co_probs, type = "l",)
abline(h = 1 - alpha, col = "red", lty = 2)
```

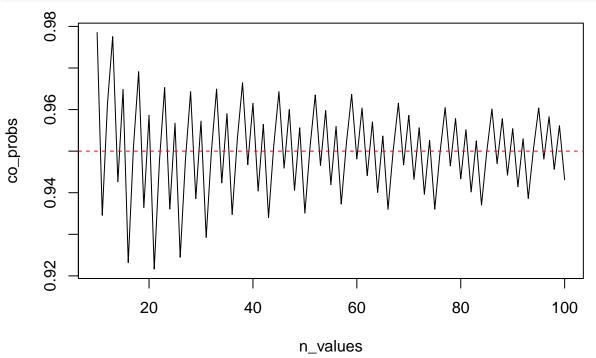


```
wilson_co_prob <- function(n, p, alpha) {</pre>
    k \leftarrow 0:n
    int <- binom.wilson(k, n, conf.level = 1 - alpha)</pre>
    co <- sum(dbinom(k, n, p) * (int$lower <= p & p <= int$upper))</pre>
    return(co)
}
ac_co_prob <- function(n, p, alpha) {</pre>
    k \leftarrow 0:n
    int <- binom.confint(k, n, conf.level = 1 - alpha, method = "ac")</pre>
    co <- sum(dbinom(k, n, p) * (int$lower <= p & p <= int$upper))</pre>
    return(co)
}
lrt_co_prob <- function(n, p, alpha) {</pre>
    k \leftarrow 0:n
    int <- binom.lrt(k, n, conf.level = 1 - alpha)</pre>
    co <- sum(dbinom(k, n, p) * (int$lower <= p & p <= int$upper))
    return(co)
}
co_probs <- sapply(n_values, function(n) wilson_co_prob(n, p, alpha))</pre>
ac_co_probs <- sapply(n_values, function(n) ac_co_prob(n, p, alpha))</pre>
```

lrt_co_probs <- sapply(n_values, function(n) lrt_co_prob(n, p, alpha))</pre>

```
result <- data.frame(n = n_values, p_n = round(co_probs, 3))
result_ac <- data.frame(n = n_values, p_n = round(ac_co_probs, 3))
result_lrt <- data.frame(n = n_values, p_n = round(lrt_co_probs, 3))</pre>
```

```
# Wykres dla Wilsona
plot(n_values, co_probs, type = "l")
abline(h = 1 - alpha, col = "red", lty = 2)
```



print(result)

```
##
        n
            p_n
## 1
       10 0.979
## 2
       11 0.935
## 3
       12 0.961
## 4
       13 0.978
## 5
       14 0.943
## 6
       15 0.965
## 7
       16 0.923
## 8
       17 0.951
## 9
       18 0.969
      19 0.936
## 10
## 11
       20 0.959
## 12
       21 0.922
## 13
       22 0.948
## 14
       23 0.965
       24 0.936
## 15
## 16
       25 0.957
## 17
       26 0.924
       27 0.948
## 18
## 19
       28 0.964
## 20
       29 0.939
## 21 30 0.957
```

```
## 22 31 0.929
## 23
       32 0.950
## 24
       33 0.965
## 25
       34 0.942
## 26
       35 0.959
## 27
       36 0.935
## 28
       37 0.953
## 29
       38 0.966
## 30
       39 0.947
## 31
       40 0.962
## 32
       41 0.940
       42 0.956
## 33
## 34
       43 0.934
       44 0.951
## 35
## 36
       45 0.964
## 37
       46 0.946
## 38
       47 0.960
       48 0.941
## 39
## 40
       49 0.956
       50 0.935
## 41
## 42
       51 0.951
## 43
       52 0.964
       53 0.947
## 44
## 45
       54 0.960
## 46
       55 0.942
## 47
       56 0.956
## 48
       57 0.937
## 49
       58 0.952
## 50
       59 0.964
## 51
       60 0.948
## 52
       61 0.960
## 53
       62 0.944
## 54
       63 0.957
## 55
       64 0.940
       65 0.954
## 56
## 57
       66 0.936
## 58
       67 0.950
## 59
       68 0.962
## 60
       69 0.947
       70 0.959
## 61
## 62
       71 0.943
## 63
       72 0.956
## 64
       73 0.940
## 65
       74 0.953
## 66
       75 0.936
## 67
       76 0.950
## 68
       77 0.960
## 69
       78 0.946
       79 0.958
## 70
## 71
       80 0.943
## 72
       81 0.955
```

73

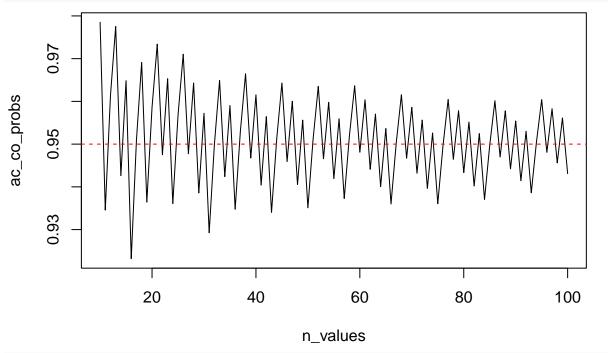
74

82 0.940 83 0.952

75 84 0.937

```
## 76 85 0.950
## 77
       86 0.960
## 78
       87 0.947
## 79
       88 0.958
       89 0.944
## 80
## 81
       90 0.955
## 82
       91 0.941
       92 0.953
## 83
## 84
       93 0.939
## 85
       94 0.951
## 86
       95 0.960
## 87
       96 0.948
## 88
       97 0.958
## 89
       98 0.946
## 90 99 0.956
## 91 100 0.943
```

```
# Wykres dla Agresti-Coull
plot(n_values, ac_co_probs, type = "1")
abline(h = 1 - alpha, col = "red", lty = 2)
```



print(result_ac)

```
##
            p_n
## 1
       10 0.979
       11 0.935
## 2
## 3
       12 0.961
## 4
       13 0.978
## 5
       14 0.943
## 6
       15 0.965
## 7
       16 0.923
## 8
       17 0.951
## 9
       18 0.969
```

```
## 10 19 0.936
## 11
      20 0.959
       21 0.973
## 12
## 13
       22 0.948
## 14
       23 0.965
## 15
       24 0.936
## 16
       25 0.957
## 17
       26 0.971
## 18
       27 0.948
## 19
       28 0.964
## 20
       29 0.939
       30 0.957
## 21
## 22
       31 0.929
## 23
       32 0.950
## 24
       33 0.965
## 25
       34 0.942
## 26
       35 0.959
## 27
       36 0.935
## 28
       37 0.953
       38 0.966
## 29
## 30
       39 0.947
## 31
       40 0.962
       41 0.940
## 32
## 33
       42 0.956
## 34
       43 0.934
## 35
       44 0.951
## 36
       45 0.964
## 37
       46 0.946
## 38
       47 0.960
## 39
       48 0.941
## 40
       49 0.956
## 41
       50 0.935
## 42
       51 0.951
## 43
       52 0.964
       53 0.947
## 44
## 45
       54 0.960
## 46
       55 0.942
## 47
       56 0.956
## 48
       57 0.937
       58 0.952
## 49
## 50
       59 0.964
## 51
       60 0.948
## 52
       61 0.960
## 53
       62 0.944
## 54
       63 0.957
## 55
       64 0.940
## 56
       65 0.954
## 57
       66 0.936
```

67 0.950

68 0.962

69 0.947

70 0.959

71 0.943

63 72 0.956

58 ## 59

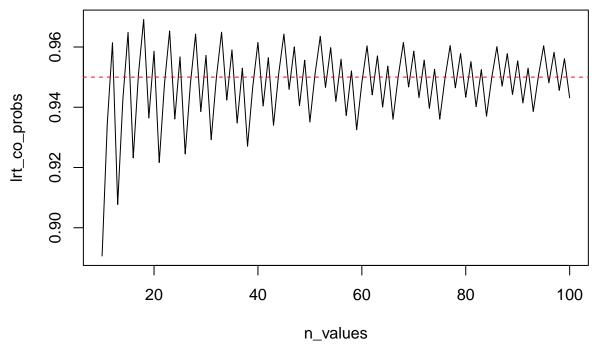
60

61

62

```
## 64
      73 0.940
## 65
       74 0.953
## 66
       75 0.936
## 67
       76 0.950
       77 0.960
## 68
##
  69
       78 0.946
## 70
       79 0.958
       80 0.943
## 71
##
  72
       81 0.955
## 73
       82 0.940
## 74
       83 0.952
## 75
       84 0.937
##
   76
       85 0.950
## 77
       86 0.960
## 78
       87 0.947
## 79
       88 0.958
## 80
       89 0.944
## 81
       90 0.955
## 82
       91 0.941
## 83
       92 0.953
       93 0.939
##
  84
## 85
       94 0.951
## 86
       95 0.960
## 87
       96 0.948
## 88
       97 0.958
## 89
       98 0.946
## 90 99 0.956
## 91 100 0.943
```

```
# Wykres dla ilorazu wiarygodności
plot(n_values, lrt_co_probs, type = "1",)
abline(h = 1 - alpha, col = "red", lty = 2)
```



print(result_lrt)

##

```
n p_n
## 1
       10 0.891
## 2
       11 0.935
## 3
       12 0.961
## 4
       13 0.908
## 5
      14 0.943
## 6
      15 0.965
## 7
       16 0.923
## 8
       17 0.951
## 9
       18 0.969
## 10 19 0.936
       20 0.959
## 11
## 12 21 0.922
## 13
      22 0.948
## 14
      23 0.965
## 15
       24 0.936
## 16
      25 0.957
## 17
      26 0.924
## 18 27 0.948
## 19
       28 0.964
## 20 29 0.939
## 21 30 0.957
## 22
      31 0.929
## 23 32 0.950
## 24 33 0.965
## 25 34 0.942
## 26
      35 0.959
## 27
       36 0.935
## 28
      37 0.953
## 29
      38 0.927
## 30
       39 0.947
## 31
      40 0.962
## 32
      41 0.940
      42 0.956
## 33
## 34
      43 0.934
## 35 44 0.951
## 36 45 0.964
## 37
      46 0.946
## 38
      47 0.960
## 39
      48 0.941
## 40 49 0.956
## 41 50 0.935
## 42 51 0.951
## 43 52 0.964
## 44
      53 0.947
       54 0.960
## 45
## 46
      55 0.942
## 47
       56 0.956
      57 0.937
## 48
## 49
       58 0.952
## 50 59 0.933
## 51 60 0.948
```

```
## 52
       61 0.960
## 53
       62 0.944
## 54
       63 0.957
## 55
       64 0.940
## 56
       65 0.954
## 57
       66 0.936
## 58
       67 0.950
## 59
       68 0.962
## 60
       69 0.947
## 61
       70 0.959
## 62
       71 0.943
## 63
       72 0.956
##
  64
       73 0.940
## 65
       74 0.953
## 66
       75 0.936
## 67
       76 0.950
## 68
       77 0.960
## 69
       78 0.946
## 70
       79 0.958
## 71
       80 0.943
## 72
       81 0.955
## 73
       82 0.940
## 74
       83 0.952
## 75
       84 0.937
## 76
       85 0.950
## 77
       86 0.960
## 78
       87 0.947
##
  79
       88 0.958
## 80
       89 0.944
## 81
       90 0.955
## 82
       91 0.941
## 83
       92 0.953
## 84
       93 0.939
## 85
       94 0.951
## 86
       95 0.960
## 87
       96 0.948
## 88
       97 0.958
## 89
       98 0.946
## 90
       99 0.956
## 91 100 0.943
```

Wykres dla ilorazu wiarygodności wydaje się mieć najmniejsze wahania