Trabajo Final - Curso: Enfoques, métodos y herramientas para el análisis de la conectividad ecológica

Que áreas prioritárias para a conservação do Pampa ocorrente no Brasil devem ter a conectividade entre elas priorizada? Uma abordagem multitáxon

José Ricardo Inacio Ribeiro

INTRODUÇÃO

Como programa integrante do Plano Plurianual (PPA) 2020-2023 – coordenado pelo MMA e estabelecido pela Lei n. 13.971, de 27/12/2019 –, a conservação e uso sustentável da biodiversidade e dos recursos naturais do Brasil visam combater e reverter as perdas e redução dos serviços ecossistêmicos por meio de políticas públicas integradoras. As áreas e ações prioritárias para a conservação, utilização sustentável e repartição dos benefícios da biodiversidade – instituídas formalmente pelo Decreto n. 5.092 de 21/05/2004, no âmbito das atribuições do Ministério do Meio Ambiente e Mudança do Clima (MMA) – são um instrumento de política pública que visa à tomada de decisão, de forma objetiva e participativa, sobre planejamento e implementação de medidas adequadas à conservação, à recuperação e ao uso sustentável de ecossistemas.

Como áreas prioritárias usualmente são definidas com base apenas em endemismos e riqueza de espécies, é mister também considerar a conectividade funcional delas porque áreas com baixa conectividade podem ter queda de biodiversidade como resultado do isolamento dessas áreas (Haddad *et al.*, 2015; Fahrig *et al.*, 2019). A conectividade da paisagem é um conceito fundamental na biologia da conservação, implicando na capacidade da paisagem de facilitar o movimento de organismos e processos ecológicos entre habitats (Taylor *et al.*, 1993). A riqueza de um fragmento também é determinada pela conectividade com outros fragmentos (Müller & Oliveira, 2020). Paisagens conectadas possuem baixa extinção local porque a biodiversidade é mantida ao longo do tempo, e a probabilidade de que os ecossistemas imersos nelas possam ser recolonizados ou reorganizados ao que eram inicialmente – antes de alguma interferência humana – aumenta. A fragmentação de habitats representa, assim, uma das principais ameaças à biodiversidade global, resultando em populações isoladas, redução do fluxo gênico e aumento do risco de extinção local (Fahrig, 2003).

No contexto do Pampa brasileiro, um dos biomas mais ameaçados do país, a avaliação da conectividade torna-se crucial para o planejamento efetivo de estratégias de conservação (Overbeck *et al.*, 2007). As 104 áreas prioritárias para a conservação (Figura 1) – instituídas formalmente pelo Decreto n. 5.092 –, assim como a atualização delas ocorrida em 2022, representam uma oportunidade única para avaliar a eficácia dessas regiões quanto à conectividade. Especialmente na Província Pampiana ocorrente no Brasil, a biodiversidade registrada nessa região vem sendo perdida em proporções alarmantes, sendo a conversão da paisagem por ação humana – em diferentes escalas – uma das principais causas desse fenômeno, especialmente o plantio de soja, silvicultura e rizicultura (Baeza *et al.*, 2022). Muito esforço de pesquisa foi empreendido em um passado recente relacionado às ações de desmatamento em florestas, mas não em ecossistemas ocorrentes em savanas, pastagens, campos e bosques abertos (Overbeck *et al.*, 2015), alguns muito presentes no Pampa brasileiro. Ainda assim, o Pampa – com sua formação predominante de pastagem – é o bioma de maior riqueza taxonômica conhecida (Overbeck *et al.*, 2007), abrigando cerca de 9% da biodiversidade brasileira (Andrade *et al.*, 2023).

Analisar múltiplas espécies com diferentes necessidades de habitat, capacidades de dispersão e sensibilidades à fragmentação oferece uma visão muito mais completa e realista da conectividade da paisagem (Ayram *et al.*, 2019; Wood *et al.*, 2022). Isso aumenta a probabilidade de que as estratégias de conservação sejam eficazes para uma gama maior da biodiversidade local. Ao comparar os padrões de conectividade de diferentes grupos, é possível identificar áreas que funcionem como "elos" para múltiplas espécies, tornando-se prioritárias para a conservação e revela áreas onde os conflitos são maiores, exigindo soluções de manejo mais complexas.

De acordo com Moreira *et al.* (2011), o primeiro e mais importante recenseamento dos representantes dos percevejos da infraordem Nepomorpha (Hemiptera: Heteroptera) – os heterópteros verdadeiramente aquáticos – no Brasil apresentou cerca de 700 espécies registradas na Região Neotropical (Polhemus & Polhemus, 2008a), pelo menos 500 delas ocorrentes no Brasil, e as famílias Belostomatidae, Corixidae, Naucoridae, Nepidae e Notonectidae mostraram-se as mais representativas. Dessas famílias, os representantes de Belostomatidae são bons exemplos de heterópteros aquáticos constituintes da comunidade bentônica, tanto em ambientes lóticos como lênticos, e podem ainda ser subcategorizados de acordo com o hábito de vida ou modo de existência (Cummins & Merritt, 1996; Polhemus, 1996; McCafferty, 1998). A família Belostomatidae é representada por insetos "escaladores" e/ou "agarradores" (Polhemus, 1996) e compreende 11 gêneros com aproximadamente 150 espécies, das quais cerca de cem estão representadas no Novo Mundo (Polhemus, 1995; Perez-Goodwyn, 2006; Estévez & Ribeiro, 2011; Ribeiro *et al.*, 2018). No que concerne ao Rio Grande do Sul, Brasil, 16 espécies compreendendo os gêneros *Belostoma* Latreille, 1807, *Horvathinia* Montandon, 1911 e *Lethocerus* Mayr, 1853 foram registradas até o momento (Moreira *et al.*, 2011). Além disso, os Belostomatidae são insetos bastante peculiares porque seus representantes compreendem um dos cinco grupos de Hemiptera onde o cuidado paternal está presente (REQUENA *et al.*, 2010). O cuidado mediado pelos machos é compartilhado provavelmente por todos os Belostomatidae (Thrasher *et al.*, 2015) e apresenta duas diferentes vias evolutivas. O cuidado do tipo *back-brooding*, quando as fêmeas põem os ovos já fecundados sobre o dorso dos machos, é provavelmente compartilhado por todos os integrantes de *Belostoma* e *Horvathinia* (Ribeiro *et al.*, 2018). Já o do tipo *emergent-brooding*, quando os machos cuidam dos ovos depositados pelas fêmeas sobre hidrófitas – acima do nível d'água (Ichikawa, 1988) – ocorre nas espécies de *Lethocerus* (Ohba & Maeda, 2017).

Apesar da ausência de consenso sobre que abordagem é mais eficiente para estimar conectividade entre fragmentos com o uso de várias espécies (Marrec *et al.*, 2020), a abordagem multitáxon para avaliação de conectividade – proposta por Ayram *et al.* (2019) – representa um avanço significativo em relação aos métodos tradicionais que focam em espécies individuais. Esse método reconhece que diferentes espécies respondem de maneira distinta às características da paisagem e que uma avaliação robusta da conectividade deve considerar múltiplos grupos taxonômicos simultaneamente (abordagem *downstream sensu* Wood *et al.*, 2022).

Com o objetivo de detectar áreas e conexões consideradas de extrema importância ambiental – permitindo que haja melhor capacidade de resiliência delas, assim como maior probabilidade de serem recolonizadas mantendo a biodiversidade ao longo do tempo – foi considerado, neste estudo, o uso de múltiplas espécies – com diferentes necessidades de habitat, capacidades de dispersão e sensibilidades à fragmentação – no processo de construção de modelos de corredores ecológicos entre essas áreas prioritárias. Diferentes cenários foram considerados levando-se em conta os tipos de cuidado paternal porque esses comportamentos alteraram indiretamente as taxas de

dispersão desses grupos durante a evolução de suas distribuições (Ribeiro *et al.*, dados não publicados). Para tanto, foram levados em consideração (1) o custo do corredor sugerido entre as áreas prioritárias, (2) a riqueza potencial média de espécies das áreas conectadas pelo corredor baseada em mapas de adequabilidade ambiental com o uso de variáveis de paisagem e (3) a áreas das manchas conectadas. O que se espera é que com os *back-brooders* haja resistência mais alta em áreas distantes de banhados e corpos d'água permanentes, enquanto com os *emergent-brooders*, uma resistência mais alta apenas em áreas próximas a habitats aquáticos de pequeno porte e com baixa diversidade. Uma abordagem multitáxon aumenta a probabilidade de que as estratégias de conservação sejam eficazes para uma gama maior da biodiversidade local e pode revelar áreas onde os conflitos são maiores, exigindo soluções de manejo mais complexas.

MATERIAL E MÉTODOS

Preparação de dados e área de estudo

Para a criação de superfícies de resistência das espécies de Belostomatidae ocorrentes na Província Pampiana do Brasil, os dados de distribuição de 16 espécies usadas neste estudo foram compilados a partir literatura especializada e bases de dados *online*, como o *the Global Biodiversity Information Facility* (*GBIF*) e a rede *species*Link. Entretanto, esses dados de ocorrência foram considerados potencialmente enviesados quando próximos a cidades. Assim, adotou-se o uso de modelagem de nicho ecológico para a obtenção de adequabilidade potencial para cada espécie, e *ensembles* de frequência para cada espécie foram obtidos com o uso do algoritmo MaxEnt – baseado em máxima entropia e com capacidade de lidar bem com dados que possuam apenas presença (Phillips *et al.*, 2006) – produzindo mapas de presença projetados para o Estado do Rio Grande do Sul com resolução de 30'. As variáveis preditoras bioclimáticas incluíram 19 variáveis bioclimáticas do *WorldClim* (Hijmans *et al.*, 2005) – além das variáveis derivadas delas próprias para organismos aquáticos (1) índice de aridez global, (2) potencial de evapotranspiração (*potential evapo-transpiration - PET*), (3) elevação, (4) declividade, (5) acúmulo de fluxo e (6) índice de umidade topográfica (*topographic wetness index – TWI*) – conforme Cano *et al.* (2018). A acurácia de cada grupo de modelos (gerados para cada espécie) foi testada pela reconstrução de 10 réplicas de *bootstrap*, em cada uma das quais os registros de ocorrência foram divididos em dados de treinamento ou calibração e dados teste ($k = 2$). A seleção de modelos foi baseada em valores de *AUC* (*Area Under the Curve*) > 0.70 (uma métrica considerada independente de limiar de decisão conforme SWETS, 1988) e *TSS* (*True Skill Statistic*) > 0.40 (uma métrica dependente de limiar de decisão). Os limiares de conversão binária foram definidos com o intuito de maximizar sensibilidade e especificidade (Liu et al., 2005).

As variáveis preditoras de paisagem foram usadas para modelagem de nicho ecológico e obtenção de riqueza potencial de heterópteros bentônicos das manchas. Para tanto, um novo conjunto de dados – incluindo representantes dos seis gêneros de heterópteros aquáticos (*Ambrysus*, *Cryphocricos*, *Limnocoris*, *Pelocoris*, *Curicta* e *Ranatra*) ocorrentes preferencialmente em fundos ou associados a superfícies de hidrófitas, rochas, gravetos e outros substratos sólidos de rios e poças localizados no Estado – foi adicionado aos registros de Belostomatidae usados, já que compreendem a maior parte da diversidade de insetos aquáticos desses ambientes (McCafferty, 1998). Assim, um mapa de riqueza – também com resolução de 30' – baseado em *ensembles* de frequência para cada espécie foi construído, com o uso de variáveis de paisagem. Para cada espécie, um grupo de mapas binários (contabilizando as réplicas), com valores de 1 para presença e 0 para ausência, projetados para o Estado, foi convertido em um mapa de adequabilidade ambiental consenso com valores de frequência média variando de 0 até 1, já contabilizando assim a incerteza na previsão (e.g., Zucchetto *et al.*, 2021). Posteriormente, esses mapas de consenso foram convertidos novamente em mapas binários, assumindo-se presença naqueles locais cujos valores foram maiores que 0,50 de frequência média. A sobreposição de todos esses mapas produziu um mapa de riqueza potencial (conforme lógica de Real *et al.*, 2017) – após as espécies com menos de 15 registros de ocorrência terem sido adicionadas –, e os valores de riqueza puderam fornecer atributos para as áreas prioritárias usadas neste estudo, funcionando como um indicador da qualidade de uso do solo e paisagem. Assim, valores altos de riqueza sugeriram melhor qualidade da mancha e valores baixos, pior qualidade da mancha. As

variáveis de paisagem basearam-se em (1) dados da área de cobertura vegetal – simplificados em três categorias de acordo com a composição dessas manchas (florestas e vegetação ripária densa, vegetação aberta e áreas úmidas e áreas antropizadas) – para a produção de mapas de conectividade estrutural, (2) índice médio de vegetação (*Enhanced Vegetation Index* - *EVI*), (3) heterogeneidade, (4) distância de rios e (5) distância de corpos d'água.

A criação de superfícies de resistência envolveu três etapas principais (adaptado de Ayram et al., 2019). A etapa 1 envolveu a soma das adequabilidades ambientais – com variáveis bioclimáticas – para cada *ensemble* de espécies referentes a cada cenário proposto. Assumindo que áreas adequadas para múltiplas espécies são mais importantes para a conservação da conectividade (Beier *et al.*, 2008), a soma representou a adequabilidade cumulativa do habitat para cada grupo taxonômico (cada cenário). Na etapa 2, a soma das adequabilidades foi escalonada linearmente para o intervalo 0-100, garantindo que os valores fossem comparáveis entre diferentes *ensembles* e facilitando a interpretação dos resultados (conforme Zeller *et al.*, 2012). A etapa 3 envolveu a conversão desses mapas de adequabilidade em resistência através da inversão dos valores dos mapas de adequabilidade. Essa conversão assume que áreas de alta adequabilidade oferecem baixa resistência ao movimento, enquanto áreas inadequadas representam alta resistência (Spear *et al.*, 2010). Valores de resistência zero foram ajustados para 0,1, a fim de evitar problemas computacionais nas análises subsequentes. Assim, as superfícies de resistência de cada grupo de espécies foram somadas – produzindo modelos escalonados entre 0,1 e 100 – para a construção de modelos de resistência cumulativos.

Cenários multitáxon

O uso de espécies com hábitos contrastantes pode resultar em diferentes propostas de corredores (Correa *et al.*, 2014), mas essa abordagem supre as necessidades diferentes que integrantes de grupos de espécies possam possuir. Além disso, esse procedimento provavelmente melhora a identificação de áreas para o planejamento de corredores (Ayrram *et al.*, 2019). Assim, 16 espécies focais de Belostomatidae – ocorrentes no Rio Grande do Sul – foram usadas para a construção dos cenários, levando-se em conta o tipo de cuidado paternal presente entre os representantes de Belostomatidae e, indiretamente, o tamanho corporal de seus representantes, já que existe uma correlação entre tamanho do corpo e tipo de cuidado paternal nessa família (Ribeiro *et al.*, dados não publicados). No cenário 1, 13 espécies de *Belostoma* e uma de *Horvathinia* foram adicionadas e se considerou uma distância de dispersão média de 82 km (Stefanello *et al.*, 2020), enquanto no cenário 2, duas espécies de *Lethocerus* e distância de dispersão 2,3 vezes maior (Ribeiro, dados não publicados). O cenário 3 combinou todas as 16 espécies, ponderado pelo número de espécies em cada um dos cenários anteriores, a fim de comparação com os outros cenários e para obter uma resposta geral da família Belostomatidae à estrutura da paisagem.

Modelos de conectividade e escolha de corredores de menor custo

A identificação de corredores de conectividade – para cada cenário – baseou-se na teoria de grafos aplicada à ecologia da paisagem, e a seleção desses corredores foi baseada na escolha de conexões com menor custo (Pinto & Keitt, 2009). Cada área prioritária foi tratada como um nó na rede, e as conexões entre nós foram determinadas pelos caminhos de custo mínimo através da superfície de resistência.

Para o cálculo das matrizes de custo mínimo e rotas de menor custo, foi usado o pacote gdistance (van Etten, 2007) do R4.5.1 (R Core Team, 2025) e envolveu a criação

de um objeto de transição baseado na superfície de resistência, a aplicação de correção geográfica para considerar a distorção da projeção e o cálculo dos custos mínimos entre todos os pares de áreas. A matriz resultante é simétrica, onde cada elemento representa o custo cumulativo de movimento entre duas áreas prioritárias (Adriaensen *et al.*, 2003).

A importância total de cada corredor foi avaliada usando o custo cumulativo dos corredores, a riqueza média de espécies das áreas conectadas pelo corredor e a área das manchas conectadas. Inicialmente, o impacto da perda de um corredor por vez foi estimado pela métrica dPC – isto é, a probabilidade de conectividade entre dois pontos aleatórios após a retirada de conexões entre elementos da paisagem (Saura & Rubio, 2010). Para tanto, a função *MK_dPCIIC_links* do pacote Makurhini (Godínez-Gómez & Ayram, 2025) do R4.5.1, com os argumentos "atribute = 'riqueza_media'", "metric = 'PC'", "removal = TRUE", "probability = 0.5" e "distance_thresholds = round(Effec_mean)" ativados. A riqueza média estimada – a ser usada como atributo das manchas na análise – foi baseada na riqueza potencial média de espécies das áreas conectadas pelo corredor, baseada em mapas de adequabilidade ambiental com o uso de variáveis de paisagem. A distância efetiva média – a ser colocada no argumento "distance_thresholds" – foi baseada no produto da média dos valores da matriz de resistência com a distância de dispersão, produzindo assim uma distância ponderada em termos de custo para cada cenário. Os corredores prioritários foram, assim, identificados selecionando-se as conexões com menor custo (percentil 95%), após estimativa da importância cumulativa baseada na média dos três critérios adotados – isto é, o impacto da perda de conectividade ao se retirar uma conexão, a riqueza potencial média de espécies das áreas conectadas pelo corredor baseada em mapas de adequabilidade ambiental com o uso de variáveis de paisagem e a área média das manchas conectadas. Essa abordagem foca nos caminhos mais viáveis para movimento de organismos, assumindo que conexões de baixo custo são mais prováveis de serem utilizadas (Rayfield *et al.*, 2011). Para cada conexão prioritária, foi calculado o caminho de custo mínimo usando o algoritmo de Dijkstra, implementado no pacote gdistance. Os caminhos resultantes representaram os corredores potenciais para movimento entre áreas prioritárias de extrema importância.

As áreas prioritárias propostas foram comparadas com o uso de índices de fragmentação em nível de mancha, a fim de entender a complexidade da forma dessas áreas indicadas pelo MMA. As manchas foram comparadas quanto à área da mancha, área-núcleo, perímetro e índice de forma (*shape index*). O índice de forma deu uma ideia geral do grau de complexidade da forma da mancha, sendo muito complexas aquelas manchas mais alongadas e pouco circulares. Para tanto, a função *MK_Fragmentation* do pacote Makurhini foi usada, com os argumentos "edge_distance = 500" e "min_node_area = 100" ativados.

RESULTADOS

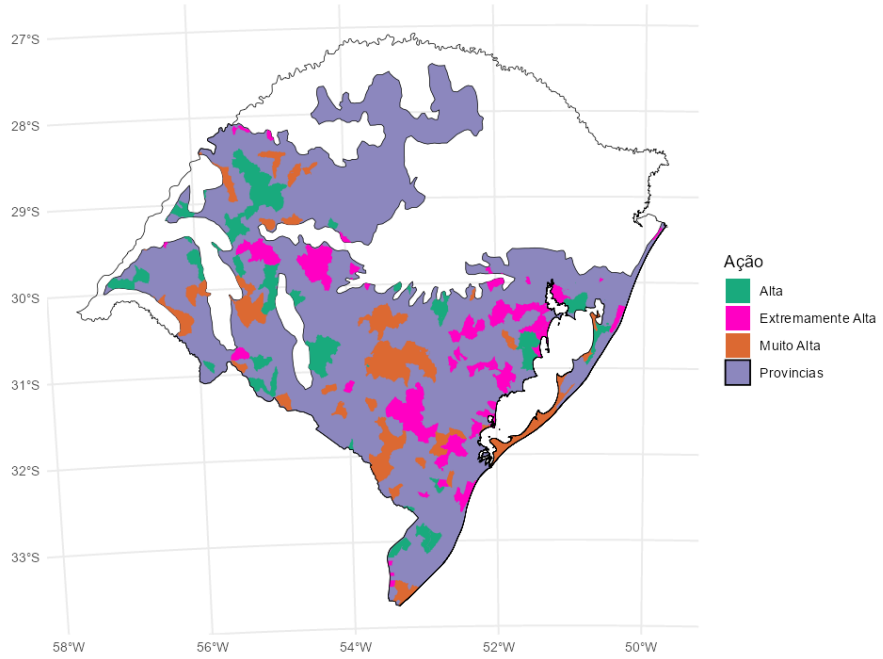

Figura 1. Mapa indicando a Província Pampiana (de acordo com Morrone *et al.*, 2014) ocorrente no Estado do Rio Grande do Sul (com cerca de 16 milhões de hectares) e as áreas prioritárias para conservação indicadas e revisadas em 2022 pelo MMA. Essas manchas foram escolhidas de acordo com as taxas de riqueza e endemismos e foram, consecutivamente, classificadas quanto ao nível de urgência para adotar alguma estratégia de conservação – isto é, alta, muito alta e extremamente alta.

O Rio Grande do Sul apresenta uma paisagem muito heterogênea formada por (1) manchas de perturbação crônica – com distúrbios originários de pastoreio –, (2) manchas introduzidas – com ambientes plantados como arrozais, cultivos de soja e eucalipto –, (3) manchas remanescentes, sendo essas duas últimas modalidades de manchas ocorrentes principalmente na metade sul do Estado, (4) corredores remanescentes e produtos de contração da rápida conversão dos ecossistemas campestres pelo plantio de culturas anuais, como a soja, e a silvicultura de espécies exóticas (principalmente *Eucalyptus* e *Acacia*) e (5) uma matriz predominantemente campestre.

Em geral, a maior frequência de área das 104 manchas (áreas prioritárias para a conservação) está em torno de 5,0 ha de unidades de $\log_{10}$. Da mesma forma, os valores mais frequentes de perímetro e área-núcleo estão em torno de 2,5 km e 4 ha de $\log_{10}$ respectivamente. Entretanto, os valores mais frequentes de índice de forma dessas manchas estão em torno de 0,2, indicando um número grande de áreas irregulares e pouco circulares. A porcentagem de fragmentação da paisagem é de 99,81%, indicando alto grau de fragmentação (Figura 2, Tabela 1).

A distribuição espacial dos corredores foi similar entre os diferentes cenários, com a importância total variando entre 0,646 e 0,696. Os modelos revelaram que a resistência inferida pela paisagem, a riqueza e a área das manchas afetam a distribuição do custo do movimento através da paisagem, principalmente no cenário 2 (nos Lethocerinae), tanto

que a maior congruência entre a distribuição dos corredores ecológicos foi entre o cenário 1 e o cenário com todas as espécies (Figuras 4-6).

Tabela 1. Sumário dos principais índices de conectividade em nível de paisagem usados neste projeto.

Métrica	Valor
Área total (ha)	4.069.261,40
Número de manchas	104
Tamanho médio dos fragmentos (ha)	39.127,51
Área-núcleo total (ha)	3.526.652,05
Índice de forma médio	0,199
Número de fragmentos menores que 100 ha	12
MESH (*Effective Mesh Size*)	30.144,53

Duzentos e sessenta e oito corredores foram considerados de extrema importância para manter as áreas prioritárias sugeridas pelo MMA com boa conectividade. Os corredores de extrema importância comum aos dois cenários concentram-se no sudoeste, no centro-oeste e numa pequena parte do nordeste do Rio Grande do Sul, onde muitos deles conectam manchas com altas quantidades de riqueza potencial baseada na paisagem. Algumas dessas áreas também foram consideradas pontes ou gargalos (habitats de passagem), ajudando no fluxo de espécies para outros ambientes adequados na paisagem.

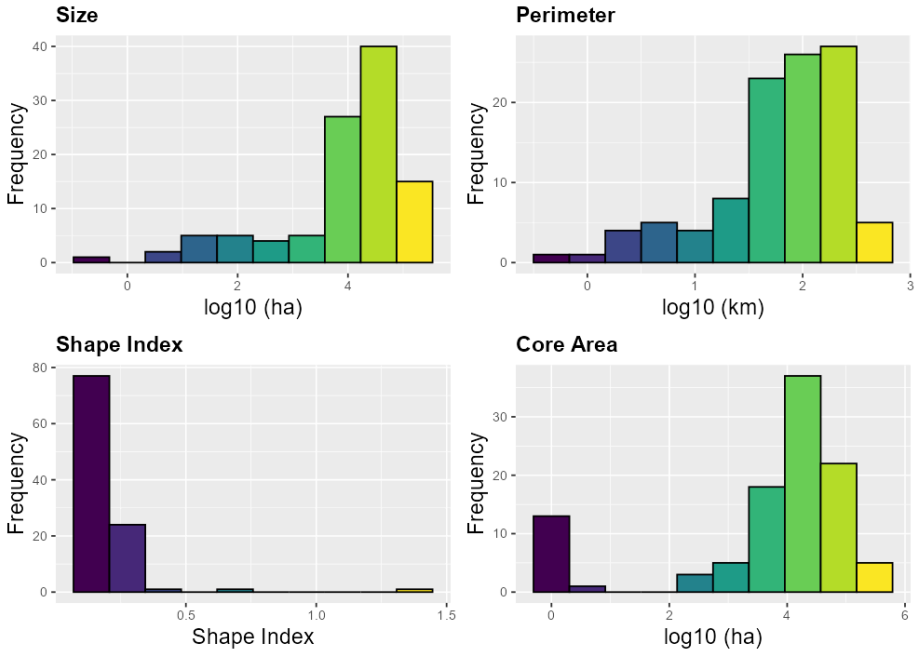

Figura 2. Histogramas indicando um sumário dos índices de conectividade em nível de paisagem das áreas prioritárias para conservação indicadas e revisadas em 2022 pelo MMA.

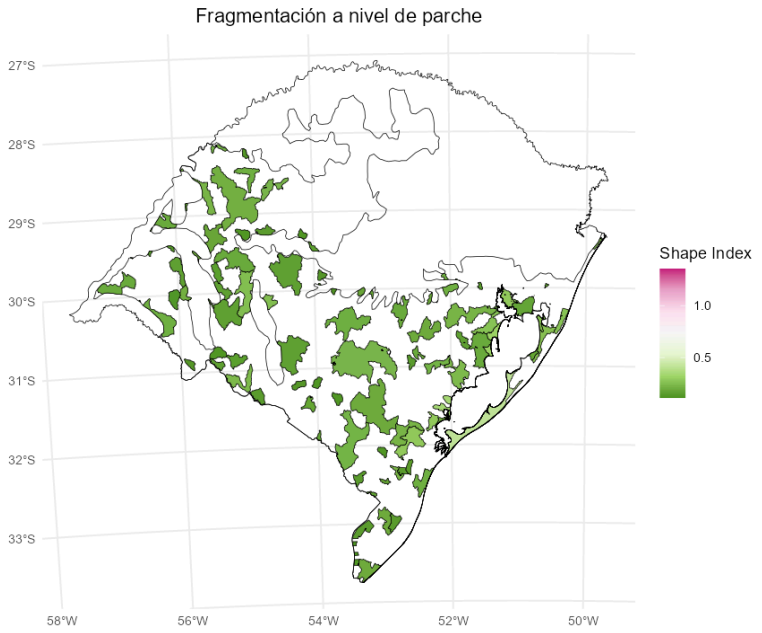

Figura 3. Mapa atribuindo valores de índice de forma para as áreas prioritárias para conservação indicadas e revisadas em 2022 pelo MMA.

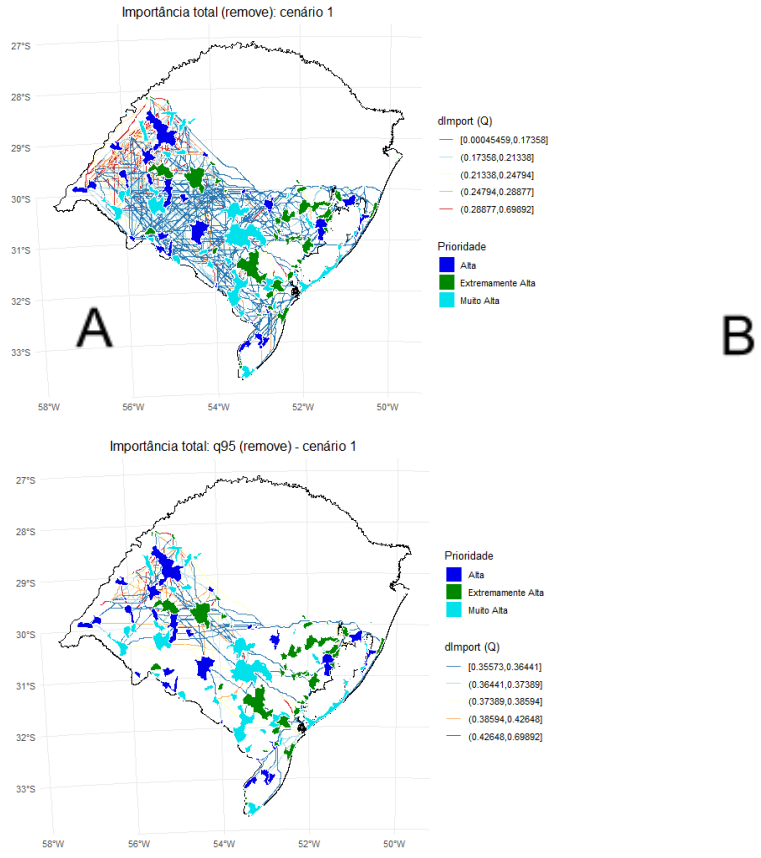

Figura 4. Mapas indicando todas as conexões avaliadas entre as 104 manchas (áreas prioritárias para a conservação) para o cenário 1 (com as espécies de *Belostoma* e *Horvathinia*). Duzentos e sessenta e oito corredores foram considerados de importância extremamente alta. A, Corredores escolhidos com base na importância total: as de cor vermelha são as que mais contribuem para manter as manchas com boa conectividade e as de cor azul-acinzentada são as que menos contribuem e precisam ser restauradas; B, corredores de extrema importância (quartil 95%) e que contribuem muito para a boa conectividade.

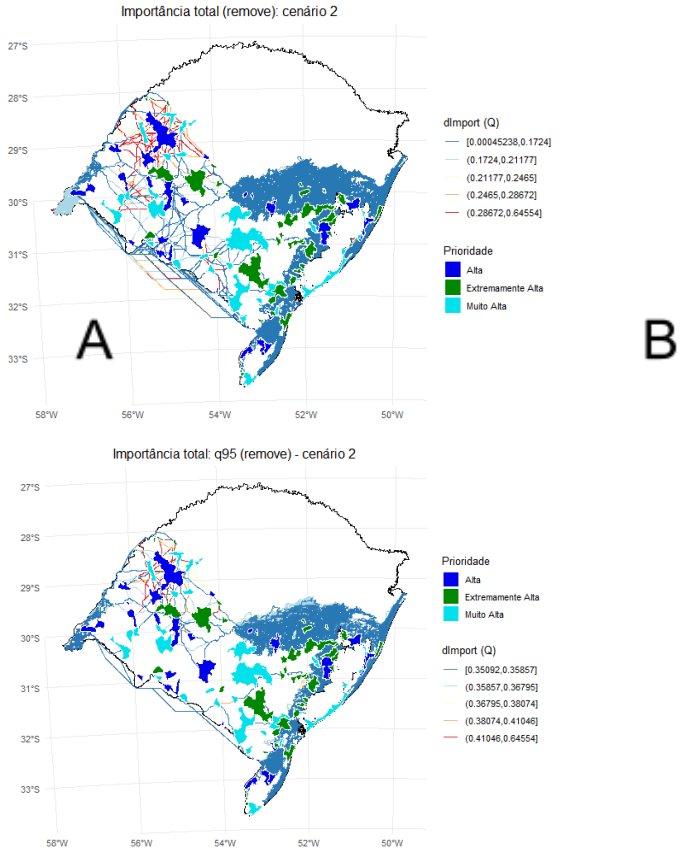

Figura 5. Mapas indicando todas as conexões avaliadas entre as 104 manchas (áreas prioritárias para a conservação) para o cenário 2 (com as espécies de *Lethocerus*). Duzentos e sessenta e oito corredores foram considerados de importância extremamente alta. A, Corredores escolhidos com base na importância total: as de cor vermelha são as que mais contribuem para manter as manchas com boa conectividade e as de cor azul-acinzentada são as que menos contribuem e precisam ser restauradas; B, corredores de extrema importância (quartil 95%) e que contribuem muito para a boa conectividade.

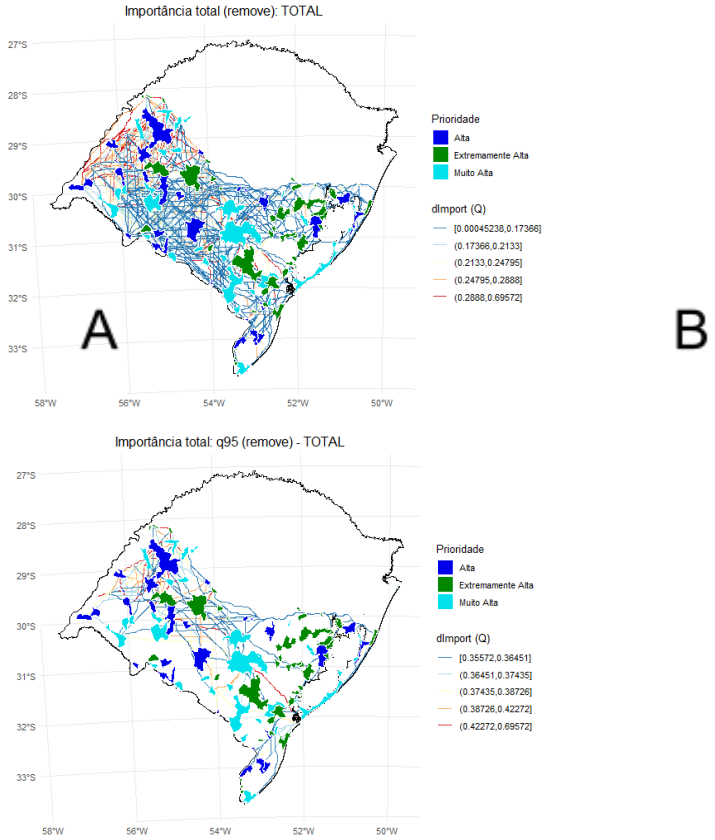

Figura 6. Mapas indicando todas as conexões avaliadas entre as 104 manchas (áreas prioritárias para a conservação) para o cenário 3 (com todas as espécies de Belostomatidae). Duzentos e sessenta e oito corredores foram considerados de importância extremamente alta. A, Corredores escolhidos com base na importância total: as de cor vermelha são as que mais contribuem para manter as manchas com boa conectividade e as de cor azul-acinzentada são as que menos contribuem e precisam ser restauradas; B, corredores de extrema importância (quartil 95%) e que contribuem muito para a boa conectividade.

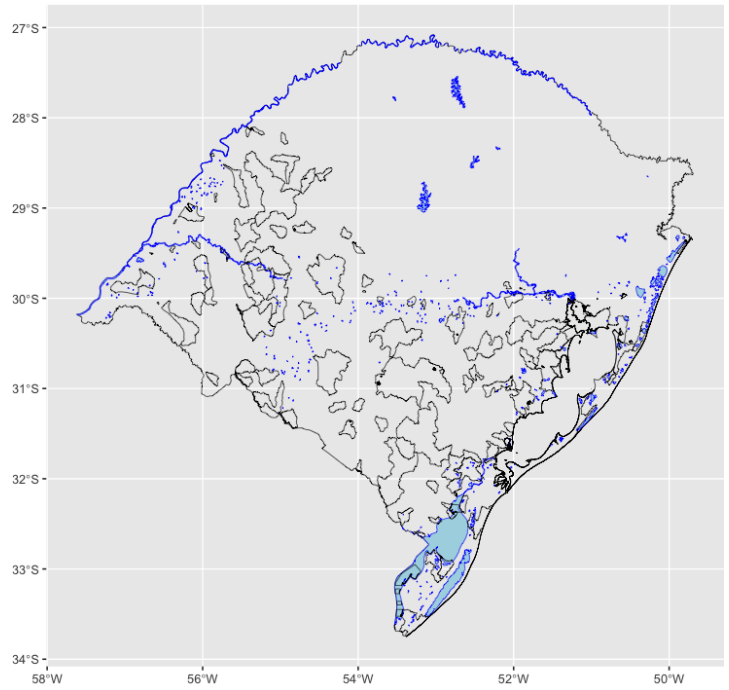

Figura 7. Mapa do Estado do Rio Grande do Sul indicando todos os corpos d'água conhecidos e disponíveis, assim como as áreas prioritárias para a conservação sugeridas pelo MMA.

No geral, um grande número de conexões de extrema importância está indicado próximas a rios e grandes lagos, principalmente aquelas do cenário 2. Entretanto, corredores de extrema importância não estão muito associados a corpos d'água de grande porte no cenário 1 (Figura 7).

DISCUSSÃO

Mesmo que o Planalto da Campanha – com altitudes de 60 a 300 m – e o Planalto Sul-Rio-Grandense – que compreende um bloco Pré-Cambriano com altitudes de 300 a 600 m – componham uma considerável parte da matriz de paisagem da Província Pampiana do Rio Grande do Sul, acredito que os elementos da paisagem que mais facilitam a dispersão dos representantes de Belostomatidae sejam os corredores – já mencionados anteriormente – frutos da contração da matriz original (corredores remanescentes de vegetação ripária), corredores ambientais com uma formação geológica de origem até Pré-cambriana (Holz & De Ros, 2000) e uma matriz de paisagem campestre, ainda que com um grau considerável de fragmentação (talvez em nível de redução e até desgaste) porque muito esforço foi dado num passado recente às ações de desmatamento em florestas, mas não em ecossistemas ocorrentes em savanas, pastagens, campos e bosques abertos (Overbeck *et al.*, 2015).

No geral, as espécies de *Lethocerus* tratadas neste projeto parecem ter maior mobilidade entre manchas pequenas e localizadas no litoral do Estado, onde muitos corpos d'água de grande porte – e até rios – estão situados. Assim, para *Lethocerus*, as áreas e conexões consideradas de extrema importância ambiental – e que oferecem

baixa resistência – quase sempre estão em áreas próximas a habitats aquáticos de grande porte, com riqueza variando entre baixa e média. Esse resultado é biologicamente coerente, pois os representantes de *Lethocerus* (*emergent-brooders*) priorizam a qualidade específica do habitat aquático sobre o tamanho das manchas. Sua maior capacidade de dispersão (2,3x) permite conectar eficientemente manchas menores, mas ecologicamente adequadas, especialmente aquelas próximas a corpos d'água de grande porte onde esses insetos podem encontrar hidrófitas necessárias para reprodução. Com o cenário 1, por outro lado, os integrantes de *Belostoma* e *Horvathinia* (*back-brooders*) mostram padrões de conectividade menos dependentes da proximidade imediata a grandes corpos d'água. Isso ocorre porque: (1) carregam os ovos no dorso, reduzindo a dependência de habitats aquáticos específicos para reprodução; (2) podem utilizar uma gama mais ampla de ambientes aquáticos temporários e permanentes; (3) sua menor capacidade de dispersão (82 km) resulta em conexões mais localizadas, priorizando a continuidade da matriz campestre sobre a proximidade a grandes rios.

A maior congruência entre o cenário 1 e o cenário 3 reflete a dominância numérica desse grupo (14 de 16 espécies, 87.5%) no *ensemble* total. Isso sugere que: (1) os padrões de conectividade do Pampa são fortemente influenciados pelas necessidades dos *back-brooders*; (2) a estratégia reprodutiva mais comum na família determina os corredores prioritários; e (3) as estratégias de conservação baseadas no cenário combinado beneficiarão principalmente as espécies de *Belostoma* e *Horvathinia*, requerendo medidas complementares específicas para *Lethocerus*.

Os corredores de extrema importância concentram-se no sudoeste, no centro-oeste e numa pequena parte do nordeste do Rio Grande do Sul, onde muitos deles conectam manchas com altas quantidades de riqueza potencial baseada na paisagem. Algumas dessas áreas também foram consideradas pontes ou gargalos (habitats de passagem), ajudando no fluxo de espécies para outros ambientes adequados na paisagem.

CONCLUSÕES

Os resultados indicam a necessidade de estratégias diferenciadas por grupo. Para os *back-brooders*, é interessante priorizar a manutenção da matriz campestre e corredores de vegetação ripária, especialmente nas regiões sudoeste e centro-oeste identificadas como críticas. Para os *emergent-brooders*, é interessante focar na proteção de corpos d'água de grande porte e suas margens, especialmente no litoral, garantindo a presença de hidrófitas emergentes.

Recomenda-se que haja validação de campo dos corredores prioritários identificados, uma análise de conectividade sazonal considerando variações nos recursos hídricos, integração com dados de movimento real das espécies e uma avaliação da efetividade dos corredores através de monitoramento populacional.

REFERÊNCIAS

Adriaensen, F., Chardon, J. P., De Blust, G., Swinnen, E., Villalba, S., Gulinck, H., & Matthysen, E. (2003). The application of 'least-cost' modelling as a functional landscape model. *Landscape and Urban Planning*, 64(4), 233-247.

Andrade, B. O., Marchesi, E., Burkart, S., Setubal, R. B., Lezama, F., Perelman, S., ... & Pillar, V. D. (2023). Vascular plant species richness and composition in the Uruguayan grasslands. *Journal of Vegetation Science*, 34(1), e13157.

Ayram, C. A. C., Mendoza, M. E., Etter, A., & Salicrup, D. R. P. (2019). Effect of the landscape matrix condition for prioritizing multispecies connectivity conservation in a highly fragmented tropical dry forest. *Environmental Management*, 64(5), 549-566.

Baeza, S., Arim, M., Cesa, A., Pereira, M., & Paruelo, J. M. (2022). Land use/land cover change (2000-2014) in the Rio de la Plata grasslands: An analysis based on MODIS NDVI time series. *Remote Sensing*, 14(3), 648.

Beier, P., Majka, D. R., & Spencer, W. D. (2008). Forks in the road: Choices in procedures for designing wildland linkages. *Conservation Biology*, 22(4), 836-851.

Cano, E., Dengler, J., Gómez-Mercado, F., Melendo, M., Valle, F., & García-Fuentes, A. (2018). Macroclimatic and soil conditions associated with the distribution of vegetation series in the southeastern Iberian Peninsula. *Applied Vegetation Science*, 21(2), 246-259.

Correa, C., Bravo, L., & Hendry, A. P. (2014). Reciprocal trophic niche shifts in native and invasive fish: Salmonids and galaxiids in Patagonian lakes. *Freshwater Biology*, 59(9), 1818-1833.

Cummins, K. W., & Merritt, R. W. (1996). Ecology and distribution of aquatic insects. In *An introduction to the aquatic insects of North America* (pp. 74-86). Kendall/Hunt Publishing.

Estévez, A. L., & Ribeiro, J. R. I. (2011). A new species of *Belostoma* Latreille from Argentina (Heteroptera: Belostomatidae). *Zootaxa*, 2965(1), 58-62.

Fahrig, L. (2003). Effects of habitat fragmentation on biodiversity. *Annual Review of Ecology, Evolution, and Systematics*, 34(1), 487-515.

Fahrig, L., Arroyo‐Rodríguez, V., Bennett, J. R., Boucher‐Lalonde, V., Cazetta, E., Currie, D. J., ... & Watling, J. I. (2019). Is habitat fragmentation bad for biodiversity? *Biological Conservation*, 230, 179-186.

Haddad, N. M., Brudvig, L. A., Clobert, J., Davies, K. F., Gonzalez, A., Holt, R. D., ... & Townshend, J. R. (2015). Habitat fragmentation and its lasting impact on Earth's ecosystems. *Science Advances*, 1(2), e1500052.

Hijmans, R. J., Cameron, S. E., Parra, J. L., Jones, P. G., & Jarvis, A. (2005). Very high resolution interpolated climate surfaces for global land areas. *International Journal of Climatology*, 25(15), 1965-1978.

Holz, M., & De Ros, L. F. (2000). Geology of the Paraná Basin (Brazil, Argentina, Paraguay and Uruguay). In *Petroleum geology of South America* (pp. 603-617). AAPG Memoir 70.

Ichikawa, N. (1988). Male brooding behavior of the giant water bug *Lethocerus deyrollei* (Hemiptera: Belostomatidae). *Journal of Ethology*, 6(2), 121-127.

Liu, C., Berry, P. M., Dawson, T. P., & Pearson, R. G. (2005). Selecting thresholds of occurrence in the prediction of species distributions. *Ecography*, 28(3), 385-393.

Marrec, R., Badenhausser, I., Bretagnolle, V., Börger, L., Roncoroni, M., Guillon, N., & Gauffre, B. (2020). Crop succession and habitat preferences drive the distribution and abundance of carabid beetles in an agricultural landscape. *Agriculture, Ecosystems & Environment*, 199, 282-289.

McCafferty, W. P. (1998). *Aquatic entomology: The fishermen's and ecologists' illustrated guide to insects and their relatives*. Jones & Bartlett Learning.

Moreira, F. F. F., Barbosa, J. F., Ribeiro, J. R. I., & Alecrim, V. P. (2011). Checklist and distribution of semiaquatic and aquatic Heteroptera (Gerromorpha and Nepomorpha) occurring in Brazil. *Zootaxa*, 2958(1), 1-74.

Morrone, J. J. (2014). Biogeographical regionalisation of the Neotropical region. *Zootaxa*, 3782(1), 1-110.

Müller, S. C., & Oliveira, J. M. (2020). Connectivity and fragmentation effects on plant communities in the Atlantic Forest. *Forest Ecology and Management*, 472, 118243.

Ohba, S. Y., & Maeda, T. (2017). Paternal care behavior and factors affecting male desertion in the giant water bug *Lethocerus deyrollei* (Hemiptera: Belostomatidae). *Entomological Science*, 20(2), 185-190.

Overbeck, G. E., Müller, S. C., Fidelis, A., Pfadenhauer, J., Pillar, V. D., Blanco, C. C., ... & Forneck, E. D. (2007). Brazil's neglected biome: The South Brazilian Campos. *Perspectives in Plant Ecology, Evolution and Systematics*, 9(2), 101-116.

Overbeck, G. E., Vélez‐Martin, E., Scarano, F. R., Lewinsohn, T. M., Fonseca, C. R., Meyer, S. T., ... & Pillar, V. D. (2015). Conservation in Brazil needs to include non‐forest ecosystems. *Diversity and Distributions*, 21(12), 1455-1460.

Perez-Goodwyn, P. J. (2006). Taxonomic revision of the subfamily Lethocerinae Lauck & Menke (Heteroptera: Belostomatidae). *Stuttgarter Beiträge zur Naturkunde Serie A*, 695, 1-71.

Phillips, S. J., Anderson, R. P., & Schapire, R. E. (2006). Maximum entropy modeling of species distributions. *Ecological Modelling*, 190(3-4), 231-259.

Pinto, N., & Keitt, T. H. (2009). Beyond the least-cost path: Evaluating corridor redundancy using a graph-theoretic approach. *Landscape Ecology*, 24(2), 253-266.

Polhemus, J. T. (1995). Family Belostomatidae Leach, 1815. In *Catalogue of the Heteroptera of the Palaearctic Region* (pp. 18-25). Netherlands Entomological Society.

Polhemus, J. T. (1996). Aquatic and semiaquatic Hemiptera. In *An introduction to the study of insects* (pp. 267-288). Saunders College Publishing.

Polhemus, J. T., & Polhemus, D. A. (2008). Global diversity of true bugs (Heteroptera; Insecta) in freshwater. *Hydrobiologia*, 595(1), 379-391.

R Core Team. (2025). *R: A language and environment for statistical computing*. R Foundation for Statistical Computing. https://www.R-project.org/

Real, R., Barbosa, A. M., Rodríguez, A., García, F. J., Vargas, J. M., Palomo, L. J., & Delibes, M. (2017). Integrating fuzzy logic in species distribution models. *Journal of Biogeography*, 44(4), 903-914.

Requena, G. S., Buzatto, B. A., Munguía‐Steyer, R., & Machado, G. (2010). Paternal care and sexual selection in arthropods. In *Animal behaviour: Evolution and mechanisms* (pp. 201-234). Springer.

Ribeiro, J. R. I., Estévez, A. L., & Moreira, F. F. F. (2018). A new species of *Belostoma* Latreille from Brazil, with a key to the species of the *B. dentatum* group (Hemiptera: Heteroptera: Belostomatidae). *Zootaxa*, 4378(3), 413-420.

Spear, S. F., Balkenhol, N., Fortin, M. J., McRae, B. H., & Scribner, K. (2010). Use of resistance surfaces for landscape genetic studies: Considerations for parameterization and analysis. *Molecular Ecology*, 19(17), 3576-3591.

Stefanello, F., Rocha, C. H., & Stenert, C. (2020). Dispersal capacity and habitat requirements of aquatic insects in temporary ponds. *Aquatic Sciences*, 82(2), 1-12.

Swets, J. A. (1988). Measuring the accuracy of diagnostic systems. *Science*, 240(4857), 1285-1293.

Taylor, P. D., Fahrig, L., Henein, K., & Merriam, G. (1993). Connectivity is a vital element of landscape structure. *Oikos*, 68(3), 571-573.

Thrasher, D. J., Reyes, L. M., & Klug, H. (2015). The evolution of parental care in insects: The roles of ecology, life history and sexual selection. In *Ecology and evolution of parental care* (pp. 251-270). Oxford University Press.

van Etten, J. (2007). R package gdistance: Distances and routes on geographical grids. *Journal of Statistical Software*, 76(13), 1-21.

Wood, K. A., Stillman, R. A., & Hilton, G. M. (2022). Multi-species approaches to conservation planning. *Conservation Biology*, 36(2), e13845.

Zeller, K. A., McGarigal, K., & Whiteley, A. R. (2012). Estimating landscape resistance to movement: A review. *Landscape Ecology*, 27(6), 777-797.

Zucchetto, M., Franzoi, P., & Pranovi, F. (2021). A ensemble modeling approach for predicting fish species distributions in data-poor environments. *Ecological Modelling*, 440, 109384.

CÓDIGO USADO

```
# projeto final: Como a conectividade dos Belostomatidae(Insecta: Hemiptera:
Heteroptera)
# se comporta entre diferentes biomas?
rm(list = ls())
pacman::p_load(raster, dismo, gam, randomForest,
               kernlab, vegan, colorRamps, sdmpredictors,
               fuzzySim, modEvA, maxnet, gbm, corrplot,
               gam, randomForest, gbm, corrplot, plotmo, ecospat,
               e1071, ENMeval, blockCV, geodata, geosphere, igraph)
library(ggplot2)
library(sf)
library(devtools)
library(remotes)
# install_github("connectscape/Makurhini", dependencies = TRUE, upgrade = "never")
library(Makurhini)
library(RColorBrewer)
library(terra)
library(devtools)
library(remotes)
library(parallel)

# Detectar número de cores (macOS)
num_cores <- parallel::detectCores()
num_cores

# Configurar processamento paralelo (usar fork no macOS)
if (Sys.info()["sysname"] == "Darwin") {  # macOS
  # Usar fork (mais eficiente no macOS)
  doParallel::registerDoParallel(cores = max(1, num_cores - 1))
  cat("Processamento paralelo configurado para macOS (fork)\n")
} else {
  # Usar PSOCK para outros sistemas
  cl <- parallel::makeCluster(max(1, num_cores - 1))
  doParallel::registerDoParallel(cl)
  cat("Processamento paralelo configurado (PSOCK)\n")
}

# ----------- PREPARANDO OS DADOS PARA A ANÁLISE ---------------------
# macos
diret1a <- '~/Dropbox/papers_books/modelagem_Belostomatidae_RS'

# windows
diret1a <- 'D:/Dropbox/papers_books/modelagem_Belostomatidae_RS'
# Asul <- vect(paste0(diret1, '/practicals/data/asul/America_Sul.shp'))

morrone <- read_sf(paste0(diret1a, '/morroneNew/NeotropicMap_Geo.shp'))
morrone$Provincias

# Pampean Province
pampaMorrone <- subset(morrone, morrone$Provincias == "Pampean province")
plot(pampaMorrone$geometry)

# server:
diretCoun <- paste0(diret1a, '/countries/BRA_adm')
# if (!file.exists(diretCoun)) dir.create(diretCoun)
estados <- read_sf(paste0(diretCoun, '/BRA_adm1.shp'))

# plot(estados, add = T, col = 'grey')
rioSul <- subset(estados, estados$NAME_1 %in% 'Rio Grande do Sul')

# 2. Antes de cortar, verifique se os dois shapefiles têm o mesmo CRS (Sistema de
Coordenadas
# de Referência) # Esta é a causa mais comum de problemas em operações espaciais.
print(st_crs(pampaMorrone))
print(st_crs(rioSul)) # Se forem diferentes, transforme um deles para corresponder ao
outro.
# Exemplo: transformando 'pampaMorrone' para o CRS de 'rioSul'
if (st_crs(pampaMorrone) != st_crs(rioSul)) { cat("Atenção: Os CRS são diferentes.
  Transformando 'pampaMorrone'...\n")
  pampaMorrone <- st_transform(pampaMorrone, crs = st_crs(rioSul))
}
cat("Verificando e corrigindo a geometria de 'pampaMorrone'...\n")
pampaMorrone_valid <- st_make_valid(pampaMorrone)
all(st_is_valid(pampaMorrone_valid))
```

```
rioSul_valid <- st_make_valid(rioSul)
all(st_is_valid(rioSul_valid))
# 3. Calcule a interseção espacial entre os dois shapefiles
# Isso vai "cortar" o pampaMorrone para manter apenas as partes que estão dentro do
rioSul.
library(rmapshaper)
mask_RS <- ms_dissolve(rioSul_valid)
pampaMorroneRS <- ms_clip(pampaMorrone_valid, mask_RS)
# pampaMorroneRS <- st_intersection(pampaMorrone_valid, rioSul_valid)
# 4. (Opcional) Verifique o resultado # Plote o resultado para ver se o corte foi
feito corretamente
plot(st_geometry(pampaMorroneRS), col = 'green', border = 'darkgreen') # Adicione o
contorno do estado para referência plot(st_geometry(rioSul), add = TRUE, border =
'blue', lwd = 2) # Verifique a tabela de atributos do novo objeto
head(pampaMorroneRS)
# plot(st_geometry(rioSul), add = TRUE, border = 'blue', lwd = 2) # Adiciona a borda
do estado
############################################################
# ------------- PAMPA ---------------------------------------
# áreas prioritárias de conservação no Pampa (macos)
areas_pampa <-
read_sf('~/Dropbox/papers_books/modelagem_Belostomatidae_RS/area_conservação/shape_pa
mpa/Pampa_2a_atualizacao.shp')
# plot(areas_pampa, add = T, col = 'blue')
# windows:
areas_pampa <-
read_sf('D:/Dropbox/papers_books/modelagem_Belostomatidae_RS/area_conservação/shape_p
ampa/Pampa_2a_atualizacao.shp')

areas_pampa_valid <- st_make_valid(areas_pampa)

# Esta é a etapa de diagnóstico. Vamos ver no que eles diferem.
st_crs(areas_pampa_valid)
st_crs(pampaMorroneRS)

# SIRGAS 2000 / UTM zone 22S é o padrão para o Sul do Brasil.
# O código EPSG é a forma mais fácil de especificá-lo.
crs_projetado_metros <- "EPSG:31982"

# 2. Transforme o seu shapefile para este CRS projetado
# paisagem
pampaMorroneRS_projetado <- st_transform(pampaMorroneRS, crs = crs_projetado_metros)
print(st_crs(pampaMorroneRS))
print(st_crs(pampaMorroneRS_projetado))
plot(pampaMorroneRS_projetado$geometry)

# paisagem
areas_pampa_valid_reprojetado <- st_transform(areas_pampa_valid, crs =
st_crs(pampaMorroneRS_projetado))


# ggplot() +
#   geom_sf(data = rioSul, aes(color = "Study area"), fill = NA, color = "black") +
#   geom_sf(data = areas_pampa, aes(color = "Parches"), fill = "forestgreen",
linewidth = 0.5) +
#   scale_color_manual(name = "", values = "black")+
#   theme_minimal() +
#   theme(axis.title.x = element_blank(),
#         axis.title.y = element_blank())

mask_pampaMorroneRS <- ms_dissolve(pampaMorroneRS_projetado)
areas_pampaRS <- ms_clip(areas_pampa_valid_reprojetado, mask_pampaMorroneRS)
# plot(areas_pampaRS$geometry)

# selecionando algumas colunas...
areas_pampaRS <- areas_pampaRS[, c('COD_area', 'Prior_acao', 'Area_ha')]

# raster attribute: raster de riqueza de espécies baseado em variáveis de paisagem
# Windows
no_attrib <-
rast('D:/Dropbox/papers_books/modelagem_Belostomatidae_RS/outputsBelostomatidae/spp_m
odels/paisagem/RESULTADOS/paisagem_30s_riqueza.tif')

# MACOS
no_attrib <-
rast('~/Dropbox/papers_books/modelagem_Belostomatidae_RS/outputsBelostomatidae/spp_mo
dels/paisagem/RESULTADOS/paisagem_30s_riqueza.tif')
```

```
no_attrib_projetado <- project(no_attrib, crs_projetado_metros)
plot(no_attrib_projetado)
plot(areas_pampaRS, add = T)
library(classInt)
library(dplyr)

# Use a função extract do terra para calcular a média da riqueza para cada polígono
# O argumento 'fun=mean' é a chave aqui. 'na.rm=TRUE' remove pixels NA do cálculo.
# O resultado será um data frame com o ID do polígono e o valor médio calculado.
atributos_agregados <- terra::extract(no_attrib_projetado, areas_pampaRS,
                          fun = mean, na.rm = TRUE)

# Agora, junte esses valores de volta ao seu objeto sf 'areas_pampa'
# O data frame 'atributos_agregados' tem uma coluna 'ID' que corresponde à ordem dos
polígonos.
# Vamos criar uma nova coluna, por exemplo, "riqueza_media"
areas_pampaRS <- areas_pampaRS %>%
  mutate(riqueza_media = atributos_agregados[,2]) # A segunda coluna tem os valores
da média

rioSul_reprojetado <- st_transform(rioSul_valid, crs = crs_projetado_metros)

# raster_map <- as(raster(no_attrib), "SpatialPixelsDataFrame")
# raster_map <- as.data.frame(raster_map)
# colnames(raster_map) <- c("value", "x", "y")

ggplot() +
  # geom_tile(data = raster_map, aes(x = x, y = y, fill = value), alpha = 0.8) +
  geom_sf(data = pampaMorroneRS_projetado, aes(fill = "Provincias"), color = "black")
+
  geom_sf(data = areas_pampaRS, aes(fill = Prior_acao), color = NA) +
  geom_sf(data = rioSul_reprojetado, aes(color = "Study area"), fill = NA, color =
"black") +
  scale_fill_manual(name = "Ação", values = c("#1DAB80", "#FF00C5", "#E06936",
"#8D8BBE"))+
  theme_minimal()

# matriz de resistência: adeaubilidade ambiental das espécies estudadas
# matrizes de resistência
### LOOP PARA APLICAR A TODAS AS ESPÉCIES ########
genusTaxa <- list()
mo_species <- list()
contata <- 0
valid_species <- NULL

generos <- c('Belostoma', 'Horvathinia', 'Lethocerus')

nepoGerro <- read.csv(file = paste0(diret1a, '/NepoGerroNovo.csv'), h = T, sep = ';',
                      dec = '.')

qde <- 0
for(i in generos){
  qde[i] <- dim(subset(nepoGerro, Genus == i))[1]
}
qdeSum <- cumsum(qde)
Belostomatidae <- matrix(NA, nr = sum(qde), nc = 4)
dim(Belostomatidae)

for(j in generos){
  # sppp <- which(unique(rownames(coords)) == taxon[j])

  ##### Se quisermos todos os espécimes JUNTOS das espécies escolhidas...
  idx <- which(names(qdeSum) == j)
  Belostomatidae[(qdeSum[idx - 1] + 1) : qdeSum[idx], 3] <-
nepoGerro[which(nepoGerro$Genus == j), 4]
  Belostomatidae[(qdeSum[idx - 1] + 1) : qdeSum[idx], 4] <-
nepoGerro[which(nepoGerro$Genus == j), 5]
  Belostomatidae[(qdeSum[idx - 1] + 1) : qdeSum[idx], 1] <- rep(j, (qdeSum[idx] -
qdeSum[idx - 1]))
  Belostomatidae[(qdeSum[idx - 1] + 1) : qdeSum[idx], 2] <-
nepoGerro[which(nepoGerro$Genus == j), 3]
}

Belostomatidae <- as.data.frame(na.exclude(Belostomatidae))
colnames(Belostomatidae) <- c('genus', 'species', 'decimalLongitude',
'decimalLatitude')
```

```
head(Belostomatidae)

for(g in generos){
  # conta <- conta + 1
  genero <- g
  cat(paste0('Genus = ', g, '\n'))
  # cat("Finalising... \n")
  #------ espécies --------
  genusTaxa[[genero]] <- subset(Belostomatidae, genus == genero)
  # dim(genusTaxa)# 2239 pontos

  # espécies que ocorrem no RS (baseado em Moreira et al., 2011):
  if(genero == 'Belostoma'){
    spp_RS <- c('Belostoma_aurivillianum', 'Belostoma_bergi', 'Belostoma_candidulum',
                'Belostoma_cummingsi', 'Belostoma_dentatum', 'Belostoma_dilatatum',
                'Belostoma_elegans', 'Belostoma_elongatum', 'Belostoma_foveolatum',
                'Belostoma_horvathi', 'Belostoma_malkini', 'Belostoma_martini',
                'Belostoma_micantulum', 'Belostoma_noualhieri', 'Belostoma_oxyurum',
                'Belostoma_triangulum')
  } else if(genero == 'Horvathinia'){
    spp_RS <- 'Horvathinia_pelocoroides'
  } else if(genero == 'Lethocerus'){
    spp_RS <- c('Lethocerus_annulipes', 'Lethocerus_maximus')
  }

  # i = spp_RS[1]

  # Filtrar espécies com pontos suficientes
  for(i in spp_RS) {
    mo_species[[i]] <- subset(genusTaxa[[genero]], species == i)
    if(nrow(mo_species[[i]]) >= 15) {
      contata <- contata + 1
      valid_species[contata] <- i
    } else {
      cat("A espécie", i, "apresentou apenas", nrow(mo_species[[i]]), "pontos e será
considerada
      de outra forma.\n")
    }
  }
}

tipo <- 'ambiental'

# MACOS:
diret_resul <- paste0('outAdequabilidade/', tipo)
if (!file.exists(diret_resul))
  dir.create(diret_resul)

# Windows;
diret_resul <- paste0('D:/Dropbox/curso_conectividade/outAdequabilidade/', tipo)
if (!file.exists(diret_resul))
  dir.create(diret_resul)

# Lista para armazenar rasters de adequabilidade
adequabilidades <- list()

# Carregar adequabilidades de cada espécie
for(i in 1:length(valid_species)) {
  especie <- valid_species[i]
  arquivo_adequabilidade <- file.path(diret_resul,
                                      paste0("/PRES_meanfreqEmsemble_", tipo, "_",
especie, ".tif"))

  if(file.exists(arquivo_adequabilidade)) {
    adequabilidades[[i]] <- rast(arquivo_adequabilidade)
    cat("Carregada adequabilidade para:", especie, "\n")
  } else {
    cat("AVISO: Arquivo não encontrado para", especie, "\n")
  }
}

# Passo 1:
# Somar todas as adequabilidades (cenário 1: back-brooders)
# Filtrar apenas elementos válidos (SpatRaster)
adequabilidades_validas <- adequabilidades[sapply(adequabilidades, function(x) {
  !is.null(x) && class(x)[1] == "SpatRaster"
})]
```

```
cat("Elementos válidos:", length(adequabilidades_validas), "de",
length(adequabilidades), "\n")

# Fazer stack apenas com elementos válidos
stack_adequabilidades_Belostoma <- rast(adequabilidades_validas[c(1:12)])
soma_adequabilidade_Belostoma <- sum(stack_adequabilidades_Belostoma, na.rm = TRUE)
plot(soma_adequabilidade_Belostoma)

# Passo 2: Escalonamento para 0-100
min_val <- global(soma_adequabilidade_Belostoma, min, na.rm = TRUE)[1,1]
max_val <- global(soma_adequabilidade_Belostoma, max, na.rm = TRUE)[1,1]

adequabilidade_escalonada_Belostoma <- ((soma_adequabilidade_Belostoma - min_val) /
(max_val - min_val)) * 100
plot(adequabilidade_escalonada_Belostoma)

# Passo 3: Converter adequabilidade em resistência
resistencia_Belostoma <- 100 - adequabilidade_escalonada_Belostoma
resistencia_Belostoma[resistencia_Belostoma <= 0] <- 0.1

resistencia_Belostoma <- project(resistencia_Belostoma, crs_projetado_metros)

# Salvar resultado
nome_arquivo <- paste0("resistencia_Belostoma", "_", tipo, ".tif")
writeRaster(resistencia_Belostoma, filename = paste0('resultados_projeto/',
            nome_arquivo), overwrite = TRUE)
resistencia_Belostoma <- rast(paste0('resultados_projeto/',
                                     nome_arquivo))

# Somar todas as adequabilidades (cenário 2: emergent-brooders)
# Fazer stack apenas com elementos válidos
stack_adequabilidades_Lethocerus <- rast(adequabilidades_validas[c(13:14)])
soma_adequabilidades_Lethocerus <- sum(stack_adequabilidades_Lethocerus, na.rm =
TRUE)
plot(soma_adequabilidades_Lethocerus)

# Passo 2: Escalonamento para 0-100
min_val <- global(soma_adequabilidades_Lethocerus, min, na.rm = TRUE)[1,1]
max_val <- global(soma_adequabilidades_Lethocerus, max, na.rm = TRUE)[1,1]

adequabilidade_escalonada_Lethocerus <- ((soma_adequabilidades_Lethocerus - min_val)
/ (max_val - min_val)) * 100
plot(adequabilidade_escalonada_Lethocerus)

# Passo 3: Converter adequabilidade em resistência
resistencia_Lethocerus <- 100 - adequabilidade_escalonada_Lethocerus
resistencia_Lethocerus[resistencia_Lethocerus <= 0] <- 0.1

resistencia_Lethocerus <- project(resistencia_Lethocerus, crs_projetado_metros)

# Salvar resultado
nome_arquivo <- paste0("resistencia_Lethocerus", "_", tipo, ".tif")
writeRaster(resistencia_Lethocerus, filename = paste0('resultados_projeto/',
          nome_arquivo), overwrite = TRUE)
resistencia_Lethocerus <- rast(paste0('resultados_projeto/',
                                     nome_arquivo))

# Somar todas as adequabilidades (cenário 3: emergent-brooders + back-brooders)
# Fazer stack apenas com elementos válidos
stack_adequabilidades_Total <- rast(adequabilidades_validas)
soma_adequabilidades_Total <- sum(stack_adequabilidades_Total, na.rm = TRUE)
# plot(stack_adequabilidades_Total)

# Passo 2: Escalonamento para 0-100
min_val <- global(soma_adequabilidades_Total, min, na.rm = TRUE)[1,1]
max_val <- global(soma_adequabilidades_Total, max, na.rm = TRUE)[1,1]

adequabilidade_escalonada_Total <- ((soma_adequabilidades_Total - min_val) / (max_val
- min_val)) * 100
plot(adequabilidade_escalonada_Total)

# Passo 3: Converter adequabilidade em resistência
resistencia_Total <- 100 - adequabilidade_escalonada_Total
resistencia_Total[resistencia_Total <= 0] <- 0.1

resistencia_Total <- project(resistencia_Total, crs_projetado_metros)
```

```
# Salvar resultado
nome_arquivo <- paste0("resistencia_Total", "_", tipo, ".tif")
writeRaster(resistencia_Total, filename = paste0('resultados_projeto/',
          nome_arquivo), overwrite = TRUE)
resistencia_Total <- rast(paste0('resultados_projeto/',
                                 nome_arquivo))


ecorreg <- 'PAMPA'

area_paisagem <- st_area(pampaMorroneRS_projetado)
area_paisagem <- unit_convert(area_paisagem, "m2", "ha")
area_paisagem # 16006842


# Estatísticas das manchas:

FragmentaVal <- MK_Fragmentation(nodes = areas_pampaRS[, 'COD_area'],
                                 edge_distance = 500,
                                 min_node_area = 100,
                                 landscape_area = area_paisagem,
                                 area_unit = "ha",
                                 perimeter_unit = "km",
                                 plot = TRUE)

# SHAPE INDEX:
ggplot() +
  geom_sf(data = pampaMorroneRS_projetado, aes(color = "Study area"), fill = NA,
color = "black") +
  geom_sf(data = FragmentaVal$`Patch statistics shapefile`, aes(fill = ShapeIndex),
color = "black", size = 0.1) +
  geom_sf(data = rioSul_reprojetado, aes(color = "Study area"), fill = NA, color =
"black") +
  scale_fill_distiller(
    palette = "PiYG",
    direction = -1,
    name = "Shape Index"
  ) +
  theme_minimal() +
  labs(
    title = "Fragmentación a nivel de parche",
    fill = "Shape Index"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )

class(FragmentaVal)
#> [1] "list"
FragmentaVal$`Summary landscape metrics (Viewer Panel)`

mesh <- as.data.frame(FragmentaVal[[1]])
mesh <- mesh[13,2]
mesh_porcentage <-  (area_paisagem - mesh) * 100 / area_paisagem
mesh_porcentage
# 99.81168%

###################################
#############################################################################
#############################################################################
# Global database of lakes, reservoirs, and wetlands (WWF)
# Carregar o GLWD (após download manual)
# Level 1 (GLWD-1) comprises the 3067 largest lakes (area ≥ 50 km2) and 654 largest
# reservoirs (storage capacity ≥ 0.5 km3) worldwide, and includes extensive attribute
# data.
# Carregue os pacotes necessários
library(sf)
library(rmapshaper) # Pacote excelente para operações de clip e dissolve

# 1. Carregue seus shapefiles (seu código está perfeito)
endereco <- '~/Dropbox/papers_books/modelagem_Belostomatidae_RS/landscape'
# Código para carregar os arquivos baixados manualmente
glwd1 <- read_sf(paste0(endereco, "/data/raw/8ark3lcpfw_GLWD_level1/glwd_1.shp"))
# Carregue os pacotes
library(sf)
```

```
library(rmapshaper)
# library(rnaturalearth)

# --- Pré-processamento: Validar Geometrias ---
# glwd1_valid <- glwd1
glwd1_valid <- st_make_valid(glwd1)
rioSul_valid <- st_make_valid(rioSul)

# Level 2 (GLWD-2) comprises permanent open water bodies with a surface area
# ≥ 0.1 km2 excluding the water bodies contained in GLWD-1
glwd2 <- read_sf(paste0(endereco, "/data/raw/65sv5l285i_GLWD_level2/glwd_2.shp"))
glwd2_valid <- st_make_valid(glwd2)

# 1. Verifique se o CRS do glwd1 está faltando
if (is.na(st_crs(glwd1_valid))) {
  cat("Atenção: O CRS de 'glwd1' está faltando!\n")
  cat("Assumindo e definindo o CRS como WGS 84 (EPSG:4326).\n")

  # DEFINA o CRS original. Isso não muda as coordenadas, apenas anexa os metadados.
  glwd1_valid <- st_set_crs(glwd1_valid, 4326)
}

# 2. Verifique se o CRS do rioSul está faltando (por segurança)
if (is.na(st_crs(rioSul_valid))) {
  # Se o CRS do seu shapefile de referência também estiver faltando, você tem um
problema maior.
  # Você precisaria descobrir e definir o CRS dele também.
  # Exemplo: rioSul_valid <- st_set_crs(rioSul_valid, 4674) # SIRGAS 2000
  stop("O CRS do shapefile de referência 'rioSul' também está faltando. Não é
possível continuar.")
}

# 3. Agora que ambos os objetos têm um CRS, transforme se for necessário.
if (st_crs(glwd1_valid) != st_crs(rioSul_valid)) {
  cat("CRS são diferentes. Transformando 'glwd1' para corresponder ao CRS de
'rioSul'.\n")

  # TRANSFORME o objeto. Isso recalcula as coordenadas para o novo sistema.
  glwd1_valid <- st_transform(glwd1_valid, crs = st_crs(rioSul_valid))
}

# 3. Corte o shapefile usando a ferramenta vetorial correta
# ms_clip(target, clip_feature) -> corta o 'target' usando os limites do
'clip_feature'
glwd1_RS <- ms_clip(glwd1_valid, rioSul_valid)

#####
# 1. Verifique se o CRS do glwd1 está faltando
if (is.na(st_crs(glwd2_valid))) {
  cat("Atenção: O CRS de 'glwd1' está faltando!\n")
  cat("Assumindo e definindo o CRS como WGS 84 (EPSG:4326).\n")

  # DEFINA o CRS original. Isso não muda as coordenadas, apenas anexa os metadados.
  glwd2_valid <- st_set_crs(glwd2_valid, 4326)
}

# 3. Agora que ambos os objetos têm um CRS, transforme se for necessário.
if (st_crs(glwd2_valid) != st_crs(rioSul_valid)) {
  cat("CRS são diferentes. Transformando 'glwd1' para corresponder ao CRS de
'rioSul'.\n")

  # TRANSFORME o objeto. Isso recalcula as coordenadas para o novo sistema.
  glwd2_valid <- st_transform(glwd2_valid, crs = st_crs(rioSul_valid))
}

# 3. Corte o shapefile usando a ferramenta vetorial correta
# ms_clip(target, clip_feature) -> corta o 'target' usando os limites do
'clip_feature'
glwd2_RS <- ms_clip(glwd2_valid, rioSul_valid)


# Graficar en ggplot2 usando las clases Jenks
ggplot() +
  geom_sf(data = rioSul_valid, fill = NA, color = "black") +
  geom_sf(data = glwd1_RS, aes(color = "LAKE_NAME"), fill = 'lightblue',
          color = "blue") +
  geom_sf(data = glwd2_RS, aes(color = "TYPE"), fill = "lightblue",
```

```
          color = "blue", size = 0.1) +
  geom_sf(data = areas_pampaRS, fill = NA, color = "black") +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )

# plot(rioSul_valid$geometry, border = "red", lwd = 2)
# plot(glwd1_RS$geometry, add = T, col = "lightblue", border = "blue")
# plot(glwd2_RS$geometry, add = T, col = "lightblue", border = "blue")

################################

# para o cenário 1:
resist1 <- resistencia_Belostoma
resist1 <- round(resist1)
plot(resist1)
# validBelostoma <- valid_species[c(1:10, 12:13)]
# -------------- Cenário 1: Belostoma ------------- #
#---- Índice de conectividade composto (CCIf) con un umbral
# de dispersión de 82,000 m -----------
# y una probabilidad de conectividad de 0.5.
CCIf_cen1 <- MK_Focal_nodes(nodes = areas_pampaRS,
                        id = "COD_area",
                        attribute = 'riqueza_media',
                        raster_attribute = NULL,
                        fun_attribute = NULL,
                        distance = list(type = "least-cost",
                                        resistance = resist1,
                                        least_cost.java = F,
                                        cores.java = 1),
                        metric = "PC",
                        probability = 0.5,
                        distance_thresholds = 82000,
                        search_buffer = 20000,
                        fragmentation = F,
                        intern = F,
                        # parallel = 2,
                        write =
paste0("D:/Dropbox/curso_conectividade/resultados_projeto/", ecorreg,
                                       "/CCIf_Belostoma_results")
)

# Salvar resultados
tabela <- data.frame(
  COD_area = CCIf_cen1$COD_area,
  Prior_acao = CCIf_cen1$Prior_acao,
  Area_ha = CCIf_cen1$Area_ha,
  riqueza_media = CCIf_cen1$riqueza_media,
  EC_PC = CCIf_cen1$EC.PC.,
  PC = CCIf_cen1$PC,
  dPC = CCIf_cen1$dPC,
  dPCintra = CCIf_cen1$dPCintra,
  dPCflux = CCIf_cen1$dPCflux,
  dPCconnector = CCIf_cen1$dPCconnector,
  IComp = CCIf_cen1$IComp
)

write.csv(tabela, paste0("D:/Dropbox/curso_conectividade/resultados_projeto/",
ecorreg, "/CCIf_Belostoma_results.csv"), row.names = FALSE)

# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(CCIf_cen1$PC, n = 9, style = "jenks")

# Crear una nueva variable categórica con los intervalos
PC <- CCIf_cen1 %>%
  mutate(dPC_q = cut(PC,
                     breaks = breaks$brks,
                     include.lowest = TRUE,
                     dig.lab = 5))

# Graficar en ggplot2 usando las clases Jenks
ggplot() +
  geom_sf(data = pampaMorroneRS_projetado, aes(fill = "Provincias"), color = "black")
+
  geom_sf(data = rioSul_reprojetado, aes(color = "Study area"), fill = NA, color =
"black") +
```

```
  geom_sf(data = PC, aes(fill = dPC_q), color = "black", size = 0.1) +
  geom_sf(data = areas_pampaRS, fill = NA, color = "black") +
  scale_fill_brewer(palette = "RdYlGn", direction = 1, name = "PC (jenks)") +
  theme_minimal() +
  labs(
    title = "PC en paisajes focales",
    fill = "PC"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )

# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(CCIf_cen1$dPC, n = 9, style = "jenks")

# Crear una nueva variable categórica con los intervalos
PC <- CCIf_cen1 %>%
  mutate(dPC_q = cut(dPC,
                     breaks = breaks$brks,
                     include.lowest = TRUE,
                     dig.lab = 5))

# Graficar en ggplot2 usando las clases Jenks
ggplot() +
  geom_sf(data = pampaMorroneRS_projetado, aes(fill = "Provincias"), color = "black")
+
  geom_sf(data = rioSul_reprojetado, aes(color = "Study area"), fill = NA, color =
"black") +
  geom_sf(data = PC, aes(fill = dPC_q), color = "black", size = 0.1) +
  geom_sf(data = areas_pampaRS, fill = NA, color = "black") +
  scale_fill_brewer(palette = "RdYlGn", direction = 1, name = "dPC (jenks)") +
  theme_minimal() +
  labs(
    title = "dPC",
    fill = "dPC"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )


# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(CCIf_cen1$dPCintra, n = 9, style = "jenks")

# Crear una nueva variable categórica con los intervalos
PC <- CCIf_cen1 %>%
  mutate(dPC_q = cut(dPCintra,
                     breaks = breaks$brks,
                     include.lowest = TRUE,
                     dig.lab = 5))

# Graficar en ggplot2 usando las clases Jenks
ggplot() +
  geom_sf(data = pampaMorroneRS_projetado, aes(fill = "Provincias"), color = "black")
+
  geom_sf(data = rioSul_reprojetado, aes(color = "Study area"), fill = NA, color =
"black") +
  geom_sf(data = PC, aes(fill = dPC_q), color = "black", size = 0.1) +
  geom_sf(data = areas_pampaRS, fill = NA, color = "black") +
  scale_fill_brewer(palette = "RdYlGn", direction = 1, name = "dPCintra (jenks)") +
  theme_minimal() +
  labs(
    title = "dPCintra",
    fill = "dPCintra"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )

# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(CCIf_cen1$dPCflux, n = 9, style = "jenks")

# Crear una nueva variable categórica con los intervalos
PC <- CCIf_cen1 %>%
```

```
  mutate(dPC_q = cut(dPCflux,
                     breaks = breaks$brks,
                     include.lowest = TRUE,
                     dig.lab = 5))

# Graficar en ggplot2 usando las clases Jenks
ggplot() +
  geom_sf(data = pampaMorroneRS_projetado, aes(fill = "Provincias"), color = "black")
+
  geom_sf(data = rioSul_reprojetado, aes(color = "Study area"), fill = NA, color =
"black") +
  geom_sf(data = PC, aes(fill = dPC_q), color = "black", size = 0.1) +
  geom_sf(data = areas_pampaRS, fill = NA, color = "black") +
  scale_fill_brewer(palette = "RdYlGn", direction = 1, name = "dPCflux (jenks)") +
  theme_minimal() +
  labs(
    title = "dPCflux",
    fill = "dPCflux"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )

# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(CCIf_cen1$dPCconnector, n = 9, style = "jenks")

# Crear una nueva variable categórica con los intervalos
PC <- CCIf_cen1 %>%
  mutate(dPC_q = cut(dPCconnector,
                     breaks = breaks$brks,
                     include.lowest = TRUE,
                     dig.lab = 5))

# Graficar en ggplot2 usando las clases Jenks
ggplot() +
  geom_sf(data = pampaMorroneRS_projetado, aes(fill = "Provincias"), color = "black")
+
  geom_sf(data = rioSul_reprojetado, aes(color = "Study area"), fill = NA, color =
"black") +
  geom_sf(data = PC, aes(fill = dPC_q), color = "black", size = 0.1) +
  geom_sf(data = areas_pampaRS, fill = NA, color = "black") +
  scale_fill_brewer(palette = "RdYlGn", direction = 1, name = "dPCconnector (jenks)")
+
  theme_minimal() +
  labs(
    title = "dPCconnector",
    fill = "dPCconnector"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )

# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(CCIf_cen1$IComp, n = 9, style = "jenks")

# Crear una nueva variable categórica con los intervalos
PC <- CCIf_cen1 %>%
  mutate(dPC_q = cut(IComp,
                     breaks = breaks$brks,
                     include.lowest = TRUE,
                     dig.lab = 5))

# Graficar en ggplot2 usando las clases Jenks
ggplot() +
  geom_sf(data = rioSul_reprojetado, aes(color = "Study area"), fill = NA, color =
"black") +
  geom_sf(data = PC, aes(fill = dPC_q), color = "black", size = 0.1) +
  geom_sf(data = areas_pampaRS, fill = NA, color = "black") +
  scale_fill_brewer(palette = "RdYlGn", direction = 1, name = "IComp (jenks)") +
  theme_minimal() +
  labs(
    title = "Índice de Conectividad Compuesto",
    fill = "IComp"
  ) +
  theme(
```

```
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )

# -------------- cenário 2: Lethocerus --------------------
# para o cenário 2:
resist2 <- resistencia_Lethocerus
resist2 <- round(resist2)
plot(resist2)
# validBelostoma <- valid_species[c(1:10, 12:13)]
# -------------- Cenário 2: Lethocerus ------------- #
#---- Índice de conectividade composto (CCIf) con un umbral
# de dispersión de 82,000 m -----------
# y una probabilidad de conectividad de 0.5.
CCIf_cen2 <- MK_Focal_nodes(nodes = areas_pampaRS,
                        id = "COD_area",
                        attribute = 'riqueza_media',
                        raster_attribute = NULL,
                        fun_attribute = NULL,
                        distance = list(type = "least-cost",
                                        resistance = resist2,
                                        least_cost.java = F,
                                        cores.java = 1),
                        metric = "PC",
                        probability = 0.5,
                        distance_thresholds = 82000 * 2.3, # sabe-se que Lethocerinae
dse dispersa 2,3 vezes mais facilmente
                        search_buffer = 20000,
                        fragmentation = F,
                        intern = F,
                        # parallel = 6,
                        write =
paste0("D:/Dropbox/curso_conectividade/resultados_projeto/", ecorreg,
                                       "/CCIf_Lethocerus_results")
)

# Salvar resultados
tabela <- data.frame(
  COD_area = CCIf_cen2$COD_area,
  Prior_acao = CCIf_cen2$Prior_acao,
  Area_ha = CCIf_cen2$Area_ha,
  riqueza_media = CCIf_cen2$riqueza_media,
  EC_PC = CCIf_cen2$EC.PC.,
  PC = CCIf_cen2$PC,
  dPC = CCIf_cen2$dPC,
  dPCintra = CCIf_cen2$dPCintra,
  dPCflux = CCIf_cen2$dPCflux,
  dPCconnector = CCIf_cen2$dPCconnector,
  IComp = CCIf_cen2$IComp
)

write.csv(tabela, paste0("D:/Dropbox/curso_conectividade/resultados_projeto/",
                         ecorreg, "/CCIf_Lethocerus_results.csv"), row.names = FALSE)

# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(CCIf_cen2$PC, n = 9, style = "jenks")

# Crear una nueva variable categórica con los intervalos
PC <- CCIf_cen2 %>%
  mutate(dPC_q = cut(PC,
                     breaks = breaks$brks,
                     include.lowest = TRUE,
                     dig.lab = 5))

# Graficar en ggplot2 usando las clases Jenks
ggplot() +
  geom_sf(data = pampaMorroneRS_projetado, aes(fill = "Provincias"), color = "black")
+
  geom_sf(data = rioSul_reprojetado, aes(color = "Study area"), fill = NA, color =
"black") +
  geom_sf(data = PC, aes(fill = dPC_q), color = "black", size = 0.1) +
  geom_sf(data = areas_pampaRS, fill = NA, color = "black") +
  scale_fill_brewer(palette = "RdYlGn", direction = 1, name = "PC (jenks)") +
  theme_minimal() +
  labs(
    title = "PC en paisajes focales",
    fill = "PC"
```

```
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )

# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(CCIf_cen2$dPC, n = 9, style = "jenks")

# Crear una nueva variable categórica con los intervalos
PC <- CCIf_cen2 %>%
  mutate(dPC_q = cut(dPC,
                     breaks = breaks$brks,
                     include.lowest = TRUE,
                     dig.lab = 5))

# Graficar en ggplot2 usando las clases Jenks
ggplot() +
  geom_sf(data = pampaMorroneRS_projetado, aes(fill = "Provincias"), color = "black")
+
  geom_sf(data = rioSul_reprojetado, aes(color = "Study area"), fill = NA, color =
"black") +
  geom_sf(data = PC, aes(fill = dPC_q), color = "black", size = 0.1) +
  geom_sf(data = areas_pampaRS, fill = NA, color = "black") +
  scale_fill_brewer(palette = "RdYlGn", direction = 1, name = "dPC (jenks)") +
  theme_minimal() +
  labs(
    title = "dPC",
    fill = "dPC"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )


# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(CCIf_cen2$dPCintra, n = 9, style = "jenks")

# Crear una nueva variable categórica con los intervalos
PC <- CCIf_cen2 %>%
  mutate(dPC_q = cut(dPCintra,
                     breaks = breaks$brks,
                     include.lowest = TRUE,
                     dig.lab = 5))

# Graficar en ggplot2 usando las clases Jenks
ggplot() +
  geom_sf(data = pampaMorroneRS_projetado, aes(fill = "Provincias"), color = "black")
+
  geom_sf(data = rioSul_reprojetado, aes(color = "Study area"), fill = NA, color =
"black") +
  geom_sf(data = PC, aes(fill = dPC_q), color = "black", size = 0.1) +
  geom_sf(data = areas_pampaRS, fill = NA, color = "black") +
  scale_fill_brewer(palette = "RdYlGn", direction = 1, name = "dPCintra (jenks)") +
  theme_minimal() +
  labs(
    title = "dPCintra",
    fill = "dPCintra"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )

# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(CCIf_cen2$dPCflux, n = 9, style = "jenks")

# Crear una nueva variable categórica con los intervalos
PC <- CCIf_cen2 %>%
  mutate(dPC_q = cut(dPCflux,
                     breaks = breaks$brks,
                     include.lowest = TRUE,
                     dig.lab = 5))

# Graficar en ggplot2 usando las clases Jenks
ggplot() +
```

```
  geom_sf(data = pampaMorroneRS_projetado, aes(fill = "Provincias"), color = "black")
+
  geom_sf(data = rioSul_reprojetado, aes(color = "Study area"), fill = NA, color =
"black") +
  geom_sf(data = PC, aes(fill = dPC_q), color = "black", size = 0.1) +
  geom_sf(data = areas_pampaRS, fill = NA, color = "black") +
  scale_fill_brewer(palette = "RdYlGn", direction = 1, name = "dPCflux (jenks)") +
  theme_minimal() +
  labs(
    title = "dPCflux",
    fill = "dPCflux"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )

# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(CCIf_cen2$dPCconnector, n = 9, style = "jenks")

# Crear una nueva variable categórica con los intervalos
PC <- CCIf_cen2 %>%
  mutate(dPC_q = cut(dPCconnector,
                     breaks = unique(breaks$brks),
                     include.lowest = TRUE,
                     dig.lab = 5))

# Graficar en ggplot2 usando las clases Jenks
ggplot() +
  geom_sf(data = pampaMorroneRS_projetado, aes(fill = "Provincias"), color = "black")
+
  geom_sf(data = rioSul_reprojetado, aes(color = "Study area"), fill = NA, color =
"black") +
  geom_sf(data = PC, aes(fill = dPC_q), color = "black", size = 0.1) +
  geom_sf(data = areas_pampaRS, fill = NA, color = "black") +
  scale_fill_brewer(palette = "RdYlGn", direction = 1, name = "dPCconnector (jenks)")
+
  theme_minimal() +
  labs(
    title = "dPCconnector",
    fill = "dPCconnector"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )

# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(CCIf_cen2$IComp, n = 9, style = "jenks")

# Crear una nueva variable categórica con los intervalos
PC <- CCIf_cen2 %>%
  mutate(dPC_q = cut(IComp,
                     breaks = breaks$brks,
                     include.lowest = TRUE,
                     dig.lab = 5))

# Graficar en ggplot2 usando las clases Jenks
ggplot() +
  geom_sf(data = rioSul_reprojetado, aes(color = "Study area"), fill = NA, color =
"black") +
  geom_sf(data = PC, aes(fill = dPC_q), color = "black", size = 0.1) +
  geom_sf(data = areas_pampaRS, fill = NA, color = "black") +
  scale_fill_brewer(palette = "RdYlGn", direction = 1, name = "IComp (jenks)") +
  theme_minimal() +
  labs(
    title = "Índice de Conectividad Compuesto",
    fill = "IComp"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )

# -------- Cenário 3: todas as espécies ---------------
# para o cenário 3:
resist3 <- resistencia_Total
```

```
resist3 <- round(resist3)

# validBelostoma <- valid_species[c(1:10, 12:13)]
# -------------- Cenário 2: Lethocerus ------------- #
#---- Índice de conectividade composto (CCIf) con un umbral
# de dispersión de 30,000 m -----------
# y una probabilidad de conectividad de 0.5.
CCIf <- MK_Focal_nodes(nodes = areas_pampaRS,
                       id = "COD_area",
                       attribute = 'riqueza_media',
                       raster_attribute = NULL,
                       fun_attribute = NULL,
                       distance = list(type = "least-cost",
                                       resistance = resist3,
                                       least_cost.java = F,
                                       cores.java = 1),
                       metric = "PC",
                       probability = 0.5,
                       distance_thresholds = 82000,
                       search_buffer = 20000,
                       fragmentation = F,
                       intern = F,
                       parallel = 6,
                       write =
paste0("~/Dropbox/curso_conectividade/resultados_projeto/", ecorreg,
                                      "/CCIf_LethoBelos_results")
)

# Salvar resultados
tabela <- data.frame(
  COD_area = CCIf$COD_area,
  Prior_acao = CCIf$Prior_acao,
  Area_ha = CCIf$Area_ha,
  riqueza_media = CCIf$riqueza_media,
  EC_PC = CCIf$EC.PC.,
  PC = CCIf$PC,
  dPC = CCIf$dPC,
  dPCintra = CCIf$dPCintra,
  dPCflux = CCIf$dPCflux,
  dPCconnector = CCIf$dPCconnector,
  IComp = CCIf$IComp
)

write.csv(tabela, paste0("~/Dropbox/curso_conectividade/resultados_projeto/",
                         ecorreg, "/CCIf_LethoBelos_results.csv"), row.names = FALSE)

# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(CCIf$PC, n = 9, style = "jenks")

# Crear una nueva variable categórica con los intervalos
PC <- CCIf %>%
  mutate(dPC_q = cut(PC,
                     breaks = breaks$brks,
                     include.lowest = TRUE,
                     dig.lab = 5))

# Graficar en ggplot2 usando las clases Jenks
ggplot() +
  geom_sf(data = rioSul_reprojetado, aes(color = "Study area"), fill = NA, color =
"black") +
  geom_sf(data = PC, aes(fill = dPC_q), color = "black", size = 0.1) +
  geom_sf(data = areas_pampaRS, fill = NA, color = "black") +
  scale_fill_brewer(palette = "RdYlGn", direction = 1, name = "PC (jenks)") +
  theme_minimal() +
  labs(
    title = "PC en paisajes focales",
    fill = "PC"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )

# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(CCIf$dPC, n = 9, style = "jenks")

# Crear una nueva variable categórica con los intervalos
```

```
PC <- CCIf %>%
  mutate(dPC_q = cut(dPC,
                     breaks = breaks$brks,
                     include.lowest = TRUE,
                     dig.lab = 5))

# Graficar en ggplot2 usando las clases Jenks
ggplot() +
  geom_sf(data = rioSul_reprojetado, aes(color = "Study area"), fill = NA, color =
"black") +
  geom_sf(data = PC, aes(fill = dPC_q), color = "black", size = 0.1) +
  geom_sf(data = areas_pampaRS, fill = NA, color = "black") +
  scale_fill_brewer(palette = "RdYlGn", direction = 1, name = "dPC (jenks)") +
  theme_minimal() +
  labs(
    title = "dPC",
    fill = "dPC"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )


# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(CCIf$dPCintra, n = 9, style = "jenks")

# Crear una nueva variable categórica con los intervalos
PC <- CCIf %>%
  mutate(dPC_q = cut(dPCintra,
                     breaks = breaks$brks,
                     include.lowest = TRUE,
                     dig.lab = 5))

# Graficar en ggplot2 usando las clases Jenks
ggplot() +
  geom_sf(data = rioSul_reprojetado, aes(color = "Study area"), fill = NA, color =
"black") +
  geom_sf(data = PC, aes(fill = dPC_q), color = "black", size = 0.1) +
  geom_sf(data = areas_pampaRS, fill = NA, color = "black") +
  scale_fill_brewer(palette = "RdYlGn", direction = 1, name = "dPCintra (jenks)") +
  theme_minimal() +
  labs(
    title = "dPCintra",
    fill = "dPCintra"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )

# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(CCIf$dPCflux, n = 9, style = "jenks")

# Crear una nueva variable categórica con los intervalos
PC <- CCIf %>%
  mutate(dPC_q = cut(dPCflux,
                     breaks = breaks$brks,
                     include.lowest = TRUE,
                     dig.lab = 5))

# Graficar en ggplot2 usando las clases Jenks
ggplot() +
  geom_sf(data = rioSul_reprojetado, aes(color = "Study area"), fill = NA, color =
"black") +
  geom_sf(data = PC, aes(fill = dPC_q), color = "black", size = 0.1) +
  geom_sf(data = areas_pampaRS, fill = NA, color = "black") +
  scale_fill_brewer(palette = "RdYlGn", direction = 1, name = "dPCflux (jenks)") +
  theme_minimal() +
  labs(
    title = "dPCflux",
    fill = "dPCflux"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )
```

```r
# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(CCIf$dPCconnector, n = 9, style = "jenks")

# Crear una nueva variable categórica con los intervalos
PC <- CCIf %>%
  mutate(dPC_q = cut(dPCconnector,
                     breaks = breaks$brks,
                     include.lowest = TRUE,
                     dig.lab = 5))

# Graficar en ggplot2 usando las clases Jenks
ggplot() +
  geom_sf(data = rioSul_reprojetado, aes(color = "Study area"), fill = NA, color =
"black") +
  geom_sf(data = PC, aes(fill = dPC_q), color = "black", size = 0.1) +
  geom_sf(data = areas_pampaRS, fill = NA, color = "black") +
  scale_fill_brewer(palette = "RdYlGn", direction = 1, name = "dPCconnector (jenks)")
+
  theme_minimal() +
  labs(
    title = "dPCconnector",
    fill = "dPCconnector"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )

# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(CCIf$IComp, n = 9, style = "jenks")

# Crear una nueva variable categórica con los intervalos
PC <- CCIf %>%
  mutate(dPC_q = cut(IComp,
                     breaks = breaks$brks,
                     include.lowest = TRUE,
                     dig.lab = 5))

# Graficar en ggplot2 usando las clases Jenks
ggplot() +
  geom_sf(data = rioSul_reprojetado, aes(color = "Study area"), fill = NA, color =
"black") +
  geom_sf(data = PC, aes(fill = dPC_q), color = "black", size = 0.1) +
  geom_sf(data = areas_pampaRS, fill = NA, color = "black") +
  scale_fill_brewer(palette = "RdYlGn", direction = 1, name = "IComp (jenks)") +
  theme_minimal() +
  labs(
    title = "Índice de Conectividad Compuesto",
    fill = "IComp"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )

# ============================================================================
# ============================================================================
# ----------------------- CENÁRIO 1: BELOSTOMATINAE ------------------
library(gdistance)  # Para cálculo de corredores
library(scales)     # Para normalização

areas_prioritarias = areas_pampaRS
superficie_resistencia = resist1
nome_cenario = 'back_brooders'

resistance_df <- as.data.frame(superficie_resistencia, xy = TRUE)
head(resistance_df)
dim(resistance_df)
colnames(resistance_df) <- c("x", "y", "value")

library(ggnewscale)

# Código corrigido:
p <- ggplot() +
  # Camada 1: Resistência (contínua)
  geom_tile(data = resistance_df, aes(x = x, y = y, fill = value), alpha = 0.8) +
```

```
  scale_fill_gradientn(
    name = "Resistência",
    colors = c("#000004FF", "#1B0C42FF", "#4B0C6BFF", "#781C6DFF",
               "#A52C60FF", "#CF4446FF", "#ED6925FF", "#FB9A06FF",
               "#F7D03CFF", "#FCFFA4FF")
  ) +

  # NOVA ESCALA DE FILL (reset)
  new_scale_fill() +

  # Camada 2: Áreas prioritárias (categórica)
  geom_sf(data = areas_pampaRS, aes(fill = Prior_acao),
          color = 'white', linewidth = 0.5) +
  scale_fill_manual(
    name = "Prioridade",
    values = c(
      "Extremamente Alta" = "green4",
      "Muito Alta" = "turquoise2",
      "Alta" = "blue2"
    ),
    na.value = "gray90"
  ) +

  # Camada 3: Contorno do estado
  geom_sf(data = rioSul_reprojetado, fill = NA, color = "black",
          linewidth = 0.4) +

  # Tema
  theme_minimal() +
  theme(
    axis.title.x = element_blank(),
    axis.title.y = element_blank(),
    legend.position = "right"
  )

print(p)

library(purrr)
# A los valores NA les asignamos un alto valor para evitar que pasen por ahí
superficie_resistencia[is.na(superficie_resistencia)] <- 1000

superficie_raster <- raster(superficie_resistencia)

# Criar objeto de transição
tr <- transition(superficie_raster, transitionFunction =
                   function(x) 1/mean(x), directions = 8)
tr <- geoCorrection(tr, type = "c", multpl = FALSE)

# Extrair centroides das áreas prioritárias
centroides <- st_centroid(areas_prioritarias, of_largest_polygon = T)
coords <- st_coordinates(centroides)

# Calcular matriz de custo mínimo
# matriz_custo <- costDistance(tr, coords)

# resistencia_Total <- project(resistencia_Total, crs_projetado_metros)

#Loop para estimar corredores entre parches
rutas_list <- list()
counter <- 1
for (i in 1:(nrow(coords) - 1)) {
  #cat(paste0(i, " de ", nrow(centroides), "\r"))
  counter <- 1
  rutas <- map_dfr((i + 1):nrow(coords), function(j){
    if(counter <= nrow(coords)){
      ruta <- shortestPath(tr, coords[i,], coords[j,], output = "SpatialLines")
      ruta <- st_as_sf(ruta); st_crs(ruta) <- crs_projetado_metros
      ruta$from <-i ; ruta$to <- j
      return(ruta)
    }
  })
  rutas_list[[i]] <- rutas
}
rutas_mc <- do.call(rbind, rutas_list)
rutas_mc

ggplot() +
```

```
  geom_sf(data = rioSul_reprojetado, fill = NA, color = "black") +
  geom_sf(data = rutas_mc, aes(color = "corredores"), color = "black", linewidth =
0.5) +
  # Camada 2: Áreas prioritárias (categórica)
  geom_sf(data = areas_pampaRS, aes(fill = Prior_acao),
          color = 'white', linewidth = 0.5) +
  scale_fill_manual(
    name = "Prioridade",
    values = c(
      "Extremamente Alta" = "green4",
      "Muito Alta" = "turquoise2",
      "Alta" = "blue2"
    ),
    na.value = "gray90"
  ) +
  theme_minimal() +
  labs(
    title = "Corredores potenciales"
  ) +
  theme(
    plot.title = element_text(hjust = 0.5)
  )

#Distancia efectiva promedio como umbral de distancia
Effec_mean <- mean(superficie_raster[], na.rm = TRUE) * 82000 # 82km


#Aplicamos la función
delta_cen1 <- MK_dPCIIC_links(nodes = areas_pampaRS,
                          attribute = 'riqueza_media',
                          area_unit = "ha",
                          distance = list(type = "least-cost",
                                          resistance = superficie_raster,
                                          least_cost.java = F,
                                          cores.java = 1),
                          removal = TRUE,
                          metric = "PC",
                          probability = 0.5,
                          distance_thresholds = round(Effec_mean),
                          parallel = 4,
                          parallel_mode = 0,
                          intern = TRUE,
                          write =
paste0("D:/Dropbox/curso_conectividade/resultados_projeto/", ecorreg,
                                        "/dPCIIC_links_Belos_results"))

head(delta_cen1)

cat("✓ Links calculados com sucesso:", nrow(delta_cen1), "links\n")

#Existen otras formas, pero crearé un nuevo ID
delta_cen1$ID_nuevo <- paste0(delta_cen1$Destination, "_", delta_cen1$Source)

#Guardo las rutas en un objeto nuevo para tener de respaldo mi vector original
rutas_mc2 <- rutas_mc
rutas_mc2$ID_nuevo <- paste0(rutas_mc2$from, "_", rutas_mc$to)

#Aplicar merge
rutas_mc2 <- merge(rutas_mc2, delta_cen1, by = "ID_nuevo")

# Adicionar informações do cenário
rutas_mc2$cenario <- nome_cenario
rutas_mc2$distance_threshold <- round(Effec_mean)

# Mostrar resumo dos links
cat("Resumo dos links:\n")
cat("- dPC médio:", round(mean(rutas_mc2$dPC_removal, na.rm = TRUE), 6), "\n")
cat("- dPC máximo:", round(max(rutas_mc2$dPC_removal, na.rm = TRUE), 6), "\n")
cat("- dPC mínimo:", round(min(rutas_mc2$dPC, na.rm = TRUE), 6), "\n")

links_makurhini = rutas_mc2

# ============================================================================
# CRITÉRIO 1: CUSTO DO LINK (baseado em dPC)
# Maior dPC = menor custo = maior importância
# ============================================================================
```

```
custo_min <- min(links_makurhini$dPC_removal, na.rm = TRUE)
custo_max <- max(links_makurhini$dPC_removal, na.rm = TRUE)

if(custo_max > custo_min) {
  links_makurhini$importancia_custo <- (links_makurhini$dPC_removal - custo_min) /
(custo_max - custo_min)
} else {
  links_makurhini$importancia_custo <- 0.5  # Valor neutro se todos iguais
}

# ============================================================================
# CRITÉRIO 2: RIQUEZA DAS ÁREAS CONECTADAS
# ============================================================================
# Extrair riqueza das áreas conectadas: uso do solo
riqueza_origem <- areas_prioritarias$riqueza_media[match(links_makurhini$Source,
1:nrow(areas_prioritarias))]
riqueza_destino <-
areas_prioritarias$riqueza_media[match(links_makurhini$Destination,
1:nrow(areas_prioritarias))]

# Tratar NAs (caso alguma área não seja encontrada)
riqueza_origem[is.na(riqueza_origem)] <- 0
riqueza_destino[is.na(riqueza_destino)] <- 0

# Calcular riqueza média da conexão
links_makurhini$riqueza_origem <- riqueza_origem
links_makurhini$riqueza_destino <- riqueza_destino
links_makurhini$riqueza_media_conexao <- (riqueza_origem + riqueza_destino) / 2

# Normalizar riqueza
riqueza_min <- min(links_makurhini$riqueza_media_conexao, na.rm = TRUE)
riqueza_max <- max(links_makurhini$riqueza_media_conexao, na.rm = TRUE)

if(riqueza_max > riqueza_min) {
  links_makurhini$importancia_riqueza <- (links_makurhini$riqueza_media_conexao -
riqueza_min) / (riqueza_max - riqueza_min)
} else {
  links_makurhini$importancia_riqueza <- 0.5
}

# ============================================================================
# CRITÉRIO 3: ÁREA DAS PATCHES CONECTADAS
# ============================================================================
# Extrair áreas das patches
area_origem <- areas_prioritarias$Area_ha[match(links_makurhini$Source,
1:nrow(areas_prioritarias))]
area_destino <- areas_prioritarias$Area_ha[match(links_makurhini$Destination,
1:nrow(areas_prioritarias))]

# Tratar NAs
area_origem[is.na(area_origem)] <- 0
area_destino[is.na(area_destino)] <- 0

# Calcular área média da conexão
links_makurhini$area_origem <- area_origem
links_makurhini$area_destino <- area_destino
links_makurhini$area_media_conexao <- (area_origem + area_destino) / 2

# Normalizar área
area_min <- min(links_makurhini$area_media_conexao, na.rm = TRUE)
area_max <- max(links_makurhini$area_media_conexao, na.rm = TRUE)

if(area_max > area_min) {
  links_makurhini$importancia_area <- (links_makurhini$area_media_conexao - area_min)
/ (area_max - area_min)
} else {
  links_makurhini$importancia_area <- 0.5
}

# ============================================================================
# IMPORTÂNCIA CUMULATIVA (média dos 3 critérios)
# ============================================================================
# Média aritmética dos 3 critérios
links_makurhini$importancia_total <- (links_makurhini$importancia_custo +
                                      links_makurhini$importancia_riqueza +
                                      links_makurhini$importancia_area) / 3
```

```
# Classificar por importância (maior = mais importante)
links_makurhini <- links_makurhini[order(links_makurhini$importancia_total,
decreasing = TRUE), ]
links_makurhini$rank_importancia <- 1:nrow(links_makurhini)

# Criar categorias de importância
q95 <- quantile(links_makurhini$importancia_total, 0.95, na.rm = TRUE)
q75 <- quantile(links_makurhini$importancia_total, 0.75, na.rm = TRUE)
q50 <- quantile(links_makurhini$importancia_total, 0.50, na.rm = TRUE)
q25 <- quantile(links_makurhini$importancia_total, 0.25, na.rm = TRUE)

links_makurhini$categoria_importancia <- case_when(
  links_makurhini$importancia_total >= q95 ~ "Extremamente Alta",
  links_makurhini$importancia_total >= q75 ~ "Muito Alta",
  links_makurhini$importancia_total >= q50 ~ "Alta",
  links_makurhini$importancia_total >= q25 ~ "Moderada",
  TRUE ~ "Baixa"
)
print(table(links_makurhini$categoria_importancia))

# Salvar resultados
st_write(links_makurhini,
"D:/Dropbox/curso_conectividade/resultados_projeto/PAMPA/corredores_importantes_cenar
io1.shp",
         delete_dsn = TRUE)

# Salvar tabela CSV
tabela_corredores_cen1 <- st_drop_geometry(links_makurhini)
write.csv(tabela_corredores_cen1,

"D:/Dropbox/curso_conectividade/resultados_projeto/PAMPA/corredores_importantes_cenar
io1.csv", row.names = FALSE)

if(nrow(links_makurhini) > 0) {
  cat("Link mais importante:\n")
  cat("- Origem:", links_makurhini$from[1], "→ Destino:", links_makurhini$to[1],
"\n")
  cat("- Importância total:", round(links_makurhini$importancia_total[1], 3), "\n")
  cat("- dPC:", round(links_makurhini$dPC_removal[1], 6), "\n")
  cat("- Riqueza média:", round(links_makurhini$riqueza_media_conexao[1], 2), "\n")
  cat("- Área média:", round(links_makurhini$area_media_conexao[1], 2), "ha\n")
}

# Salvar apenas os mais importantes (top 25%)
top_links_cen1 <- links_makurhini[links_makurhini$categoria_importancia %in%
c("Extremamente Alta"), ]
write.csv(top_links_cen1,
"D:/Dropbox/curso_conectividade/resultados_projeto/PAMPA/top_links_cenario1.csv",
row.names = FALSE)

resultados_finais <- list()
resultados_finais$cenario1 <- links_makurhini

subtitulo <- paste("Links de importância -",
          nrow(top_links_cen1), "de", nrow(links_makurhini), "links")
subtitulo
head(top_links_cen1)

# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(links_makurhini$dPC_removal, n = 5, style =
"quantile")

# Crear una nueva variable categórica con los intervalos
links_makurhini <- links_makurhini %>%
  mutate(dPC_q = cut(dPC_removal,
                     breaks = unique(breaks$brks),
                     include.lowest = TRUE,
                     dig.lab = 5))

# Criar mapa
p1 <- ggplot() +
  # Fundo (estado/região)
  geom_sf(data = rioSul_reprojetado, fill = NA, color = "black",
          size = 0.5) +
  geom_sf(data = links_makurhini, aes(color = dPC_q),
          size = 0.2, alpha = 0.9, linewidth = 0.5) +
  scale_color_brewer(palette = "RdYlBu", direction = -1, name = "dPC remove (Q)") +
```

```
  # Camada 2: Áreas prioritárias (categórica)
  geom_sf(data = areas_pampaRS, aes(fill = Prior_acao),
          color = 'white', linewidth = 0.5) +
  scale_fill_manual(
    name = "Prioridade",
    values = c(
      "Extremamente Alta" = "green4",
      "Muito Alta" = "turquoise2",
      "Alta" = "blue2"
    ),
    na.value = "gray90"
  ) +
  theme_minimal() +
  labs(
    title = "Priorização dos corredores (remoção): cenário 1",
    fill = "dPC"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )
p1


# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(links_makurhini$importancia_total, n = 5, style =
"quantile")

# Crear una nueva variable categórica con los intervalos
links_makurhini <- links_makurhini %>%
  mutate(import_q = cut(importancia_total,
                        breaks = unique(breaks$brks),
                        include.lowest = TRUE,
                        dig.lab = 5))

# Criar mapa
p1 <- ggplot() +
  # Fundo (estado/região)
  geom_sf(data = rioSul_reprojetado, fill = NA, color = "black",
          size = 0.5) +
  geom_sf(data = links_makurhini, aes(color = import_q),
          size = 0.2, alpha = 0.9, linewidth = 0.3) +
  scale_color_brewer(palette = "RdYlBu", direction = -1, name = "dImport (Q)") +
  # Camada 2: Áreas prioritárias (categórica)
  geom_sf(data = areas_pampaRS, aes(fill = Prior_acao),
          color = 'white', linewidth = 0.5) +
  scale_fill_manual(
    name = "Prioridade",
    values = c(
      "Extremamente Alta" = "green4",
      "Muito Alta" = "turquoise2",
      "Alta" = "blue2"
    ),
    na.value = "gray90"
  ) +
  theme_minimal() +
  labs(
    title = "Importância total (remove): cenário 1",
    fill = "dImport"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )
p1

# ---------- Top links ----------
# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(top_links_cen1$dPC_removal, n = 5, style =
"quantile")

# Crear una nueva variable categórica con los intervalos
top_links_cen1 <- top_links_cen1 %>%
  mutate(dPC_q = cut(dPC_removal,
                     breaks = unique(breaks$brks),
                     include.lowest = TRUE,
                     dig.lab = 5))
```

```r
# Criar mapa
p1 <- ggplot() +
  # Fundo (estado/região)
  geom_sf(data = rioSul_reprojetado, fill = NA, color = "black",
          size = 0.5) +
  geom_sf(data = top_links_cen1, aes(color = dPC_q),
          size = 0.2, alpha = 0.9, linewidth = 0.5) +
  scale_color_brewer(palette = "RdYlBu", direction = -1, name = "dPC remove (Q)") +
  # Camada 2: Áreas prioritárias (categórica)
  geom_sf(data = areas_pampaRS, aes(fill = Prior_acao),
          color = 'white', linewidth = 0.5) +
  scale_fill_manual(
    name = "Prioridade",
    values = c(
      "Extremamente Alta" = "green4",
      "Muito Alta" = "turquoise2",
      "Alta" = "blue2"
    ),
    na.value = "gray90"
  ) +
  theme_minimal() +
  labs(
    title = "Priorización de enlaces (remove)",
    fill = "dPC"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )
p1


# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(top_links_cen1$importancia_total, n = 5, style =
"quantile")

# Crear una nueva variable categórica con los intervalos
top_links_cen1 <- top_links_cen1 %>%
  mutate(import_q = cut(importancia_total,
                        breaks = unique(breaks$brks),
                        include.lowest = TRUE,
                        dig.lab = 5))

# Criar mapa
p1 <- ggplot() +
  # Fundo (estado/região)
  geom_sf(data = rioSul_reprojetado, fill = NA, color = "black",
          size = 0.5) +
  geom_sf(data = top_links_cen1, aes(color = import_q),
          size = 0.2, alpha = 0.9, linewidth = 0.3) +
  scale_color_brewer(palette = "RdYlBu", direction = -1, name = "dImport (Q)") +
  # Camada 2: Áreas prioritárias (categórica)
  geom_sf(data = areas_pampaRS, aes(fill = Prior_acao),
          color = 'white', linewidth = 0.5) +
  scale_fill_manual(
    name = "Prioridade",
    values = c(
      "Extremamente Alta" = "green4",
      "Muito Alta" = "turquoise2",
      "Alta" = "blue2"
    ),
    na.value = "gray90"
  ) +
  theme_minimal() +
  labs(
    title = "Importância total: q95 (remove) - cenário 1",
    fill = "dImport"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )
p1

# ---------------------- CENÁRIO 2: LETHOCERINAE ------------------
library(gdistance)  # Para cálculo de corredores
```

```
library(scales)     # Para normalização

areas_prioritarias = areas_pampaRS
superficie_resistencia = resist2
nome_cenario = 'emergent-brooders'

resistance_df <- as.data.frame(superficie_resistencia, xy = TRUE)
head(resistance_df)
dim(resistance_df)
colnames(resistance_df) <- c("x", "y", "value")

library(ggnewscale)

# Código corrigido:
p <- ggplot() +
  # Camada 1: Resistência (contínua)
  geom_tile(data = resistance_df, aes(x = x, y = y, fill = value), alpha = 0.8) +
  scale_fill_gradientn(
    name = "Resistência",
    colors = c("#000004FF", "#1B0C42FF", "#4B0C6BFF", "#781C6DFF",
               "#A52C60FF", "#CF4446FF", "#ED6925FF", "#FB9A06FF",
               "#F7D03CFF", "#FCFFA4FF")
  ) +

  # NOVA ESCALA DE FILL (reset)
  new_scale_fill() +

  # Camada 2: Áreas prioritárias (categórica)
  geom_sf(data = areas_pampaRS, aes(fill = Prior_acao),
          color = 'white', linewidth = 0.5) +
  scale_fill_manual(
    name = "Prioridade",
    values = c(
      "Extremamente Alta" = "green4",
      "Muito Alta" = "turquoise2",
      "Alta" = "blue2"
    ),
    na.value = "gray90"
  ) +

  # Camada 3: Contorno do estado
  geom_sf(data = rioSul_reprojetado, fill = NA, color = "black",
          linewidth = 0.4) +

  # Tema
  theme_minimal() +
  theme(
    axis.title.x = element_blank(),
    axis.title.y = element_blank(),
    legend.position = "right"
  )

print(p)

library(purrr)
# A los valores NA les asignamos un alto valor para evitar que pasen por ahí
superficie_resistencia[is.na(superficie_resistencia)] <- 1000

superficie_raster <- raster(superficie_resistencia)

# Criar objeto de transição
tr <- transition(superficie_raster, transitionFunction =
                   function(x) 1/mean(x), directions = 8)
tr <- geoCorrection(tr, type = "c", multpl = FALSE)

# Extrair centroides das áreas prioritárias
centroides <- st_centroid(areas_prioritarias, of_largest_polygon = T)
coords <- st_coordinates(centroides)

# Calcular matriz de custo mínimo
# matriz_custo <- costDistance(tr, coords)

# resistencia_Total <- project(resistencia_Total, crs_projetado_metros)

#Loop para estimar corredores entre parches
rutas_list <- list()
counter <- 1
```

```
for (i in 1:(nrow(coords) - 1)) {
  #cat(paste0(i, " de ", nrow(centroides), "\r"))
  counter <- 1
  rutas <- map_dfr((i + 1):nrow(coords), function(j){
    if(counter <= nrow(coords)){
      ruta <- shortestPath(tr, coords[i,], coords[j,], output = "SpatialLines")
      ruta <- st_as_sf(ruta); st_crs(ruta) <- crs_projetado_metros
      ruta$from <-i ; ruta$to <- j
      return(ruta)
    }
  })
  rutas_list[[i]] <- rutas
}
rutas_mc_letho <- do.call(rbind, rutas_list)
rutas_mc_letho

ggplot() +
  geom_sf(data = rioSul_reprojetado, fill = NA, color = "black") +
  geom_sf(data = rutas_mc_letho, aes(color = "corredores"), color = "black",
linewidth = 0.5) +
  # Camada 2: Áreas prioritárias (categórica)
  geom_sf(data = areas_pampaRS, aes(fill = Prior_acao),
          color = 'white', linewidth = 0.5) +
  scale_fill_manual(
    name = "Prioridade",
    values = c(
      "Extremamente Alta" = "green4",
      "Muito Alta" = "turquoise2",
      "Alta" = "blue2"
    ),
    na.value = "gray90"
  ) +
  theme_minimal() +
  labs(
    title = "Corredores potenciales"
  ) +
  theme(
    plot.title = element_text(hjust = 0.5)
  )

#Distancia efectiva promedio como umbral de distancia
Effec_mean <- mean(superficie_raster[], na.rm = TRUE) * 82000 * 2.3 # 82km x 2.3
vezes mais dispersão

#Aplicamos la función
delta_cen2 <- MK_dPCIIC_links(nodes = areas_pampaRS,
                              attribute = 'riqueza_media',
                              area_unit = "ha",
                              distance = list(type = "least-cost",
                                              resistance = superficie_raster,
                                              least_cost.java = F,
                                              cores.java = 1),
                              removal = TRUE,
                              metric = "PC",
                              probability = 0.5,
                              distance_thresholds = round(Effec_mean),
                              parallel = 4,
                              parallel_mode = 0,
                              intern = TRUE,
                              write =
paste0("D:/Dropbox/curso_conectividade/resultados_projeto/", ecorreg,
                                           "/dPCIIC_links_Letho_results"))

head(delta_cen2)

cat("✓ Links calculados com sucesso:", nrow(delta_cen2), "links\n")

#Existen otras formas, pero crearé un nuevo ID
delta_cen2$ID_nuevo <- paste0(delta_cen2$Destination, "_", delta_cen2$Source)

#Guardo las rutas en un objeto nuevo para tener de respaldo mi vector original
rutas_mc2_letho <- rutas_mc_letho
rutas_mc2_letho$ID_nuevo <- paste0(rutas_mc2_letho$from, "_", rutas_mc_letho$to)

#Aplicar merge
rutas_mc2_letho <- merge(rutas_mc2_letho, delta_cen2, by = "ID_nuevo")
```

```
# Adicionar informações do cenário
rutas_mc2_letho$cenario <- nome_cenario
rutas_mc2_letho$distance_threshold <- round(Effec_mean)

# Mostrar resumo dos links
cat("Resumo dos links:\n")
cat("- dPC médio:", round(mean(rutas_mc2_letho$dPC_removal, na.rm = TRUE), 6), "\n")
cat("- dPC máximo:", round(max(rutas_mc2_letho$dPC_removal, na.rm = TRUE), 6), "\n")
cat("- dPC mínimo:", round(min(rutas_mc2_letho$dPC, na.rm = TRUE), 6), "\n")

links_makurhini = rutas_mc2_letho

# ============================================================================
# CRITÉRIO 1: CUSTO DO LINK (baseado em dPC)
# Maior dPC = menor custo = maior importância
# ============================================================================

custo_min <- min(links_makurhini$dPC_removal, na.rm = TRUE)
custo_max <- max(links_makurhini$dPC_removal, na.rm = TRUE)

if(custo_max > custo_min) {
  links_makurhini$importancia_custo <- (links_makurhini$dPC_removal - custo_min) /
(custo_max - custo_min)
} else {
  links_makurhini$importancia_custo <- 0.5  # Valor neutro se todos iguais
}

# ============================================================================
# CRITÉRIO 2: RIQUEZA DAS ÁREAS CONECTADAS
# ============================================================================
# Extrair riqueza das áreas conectadas
riqueza_origem <- areas_prioritarias$riqueza_media[match(links_makurhini$Source,
1:nrow(areas_prioritarias))]
riqueza_destino <-
areas_prioritarias$riqueza_media[match(links_makurhini$Destination,
1:nrow(areas_prioritarias))]

# Tratar NAs (caso alguma área não seja encontrada)
riqueza_origem[is.na(riqueza_origem)] <- 0
riqueza_destino[is.na(riqueza_destino)] <- 0

# Calcular riqueza média da conexão
links_makurhini$riqueza_origem <- riqueza_origem
links_makurhini$riqueza_destino <- riqueza_destino
links_makurhini$riqueza_media_conexao <- (riqueza_origem + riqueza_destino) / 2

# Normalizar riqueza
riqueza_min <- min(links_makurhini$riqueza_media_conexao, na.rm = TRUE)
riqueza_max <- max(links_makurhini$riqueza_media_conexao, na.rm = TRUE)

if(riqueza_max > riqueza_min) {
  links_makurhini$importancia_riqueza <- (links_makurhini$riqueza_media_conexao -
riqueza_min) / (riqueza_max - riqueza_min)
} else {
  links_makurhini$importancia_riqueza <- 0.5
}

# ============================================================================
# CRITÉRIO 3: ÁREA DAS PATCHES CONECTADAS
# ============================================================================
# Extrair áreas das patches
area_origem <- areas_prioritarias$Area_ha[match(links_makurhini$Source,
1:nrow(areas_prioritarias))]
area_destino <- areas_prioritarias$Area_ha[match(links_makurhini$Destination,
1:nrow(areas_prioritarias))]

# Tratar NAs
area_origem[is.na(area_origem)] <- 0
area_destino[is.na(area_destino)] <- 0

# Calcular área média da conexão
links_makurhini$area_origem <- area_origem
links_makurhini$area_destino <- area_destino
links_makurhini$area_media_conexao <- (area_origem + area_destino) / 2

# Normalizar área
area_min <- min(links_makurhini$area_media_conexao, na.rm = TRUE)
```

```
area_max <- max(links_makurhini$area_media_conexao, na.rm = TRUE)

if(area_max > area_min) {
  links_makurhini$importancia_area <- (links_makurhini$area_media_conexao - area_min)
/ (area_max - area_min)
} else {
  links_makurhini$importancia_area <- 0.5
}

# ============================================================================
# IMPORTÂNCIA CUMULATIVA (média dos 3 critérios)
# ============================================================================
# Média aritmética dos 3 critérios
links_makurhini$importancia_total <- (links_makurhini$importancia_custo +
                                      links_makurhini$importancia_riqueza +
                                      links_makurhini$importancia_area) / 3

# Classificar por importância (maior = mais importante)
links_makurhini <- links_makurhini[order(links_makurhini$importancia_total,
decreasing = TRUE), ]
links_makurhini$rank_importancia <- 1:nrow(links_makurhini)

# Criar categorias de importância
q95 <- quantile(links_makurhini$importancia_total, 0.95, na.rm = TRUE)
q75 <- quantile(links_makurhini$importancia_total, 0.75, na.rm = TRUE)
q50 <- quantile(links_makurhini$importancia_total, 0.50, na.rm = TRUE)
q25 <- quantile(links_makurhini$importancia_total, 0.25, na.rm = TRUE)

links_makurhini$categoria_importancia <- case_when(
  links_makurhini$importancia_total >= q95 ~ "Extremamente Alta",
  links_makurhini$importancia_total >= q75 ~ "Muito Alta",
  links_makurhini$importancia_total >= q50 ~ "Alta",
  links_makurhini$importancia_total >= q25 ~ "Moderada",
  TRUE ~ "Baixa"
)
print(table(links_makurhini$categoria_importancia))

# Salvar resultados
st_write(links_makurhini,
"D:/Dropbox/curso_conectividade/resultados_projeto/PAMPA/corredores_importantes_cenar
io2.shp",
         delete_dsn = TRUE)

# Salvar tabela CSV
tabela_corredores_cen2 <- st_drop_geometry(links_makurhini)
write.csv(tabela_corredores_cen2,

"D:/Dropbox/curso_conectividade/resultados_projeto/PAMPA/corredores_importantes_cenar
io2.csv", row.names = FALSE)

if(nrow(links_makurhini) > 0) {
  cat("Link mais importante:\n")
  cat("- Origem:", links_makurhini$from[1], "→ Destino:", links_makurhini$to[1],
"\n")
  cat("- Importância total:", round(links_makurhini$importancia_total[1], 3), "\n")
  cat("- dPC:", round(links_makurhini$dPC_removal[1], 6), "\n")
  cat("- Riqueza média:", round(links_makurhini$riqueza_media_conexao[1], 2), "\n")
  cat("- Área média:", round(links_makurhini$area_media_conexao[1], 2), "ha\n")
}

# Salvar apenas os mais importantes (top 25%)
top_links_cen2 <- links_makurhini[links_makurhini$categoria_importancia %in%
c("Extremamente Alta"), ]
write.csv(top_links_cen2,
"D:/Dropbox/curso_conectividade/resultados_projeto/PAMPA/top_links_cenario2.csv",
row.names = FALSE)

# resultados_finais <- list()
resultados_finais$cenario2 <- links_makurhini

subtitulo <- paste("Links de importância -",
                   nrow(top_links_cen2), "de", nrow(links_makurhini), "links")

head(top_links_cen2)

# Calcular los intervalos de Jenks para strength
```

```
breaks <- classInt::classIntervals(links_makurhini$dPC_removal, n = 5, style =
"quantile")

# Crear una nueva variable categórica con los intervalos
links_makurhini <- links_makurhini %>%
  mutate(dPC_q = cut(dPC_removal,
                     breaks = unique(breaks$brks),
                     include.lowest = TRUE,
                     dig.lab = 5))

# Criar mapa
p1 <- ggplot() +
  # Fundo (estado/região)
  geom_sf(data = rioSul_reprojetado, fill = NA, color = "black",
          size = 0.5) +
  geom_sf(data = links_makurhini, aes(color = dPC_q),
          size = 0.2, alpha = 0.9, linewidth = 0.3) +
  scale_color_brewer(palette = "RdYlBu", direction = -1, name = "dPC remove (Q)") +
  # Camada 2: Áreas prioritárias (categórica)
  geom_sf(data = areas_pampaRS, aes(fill = Prior_acao),
          color = 'white', linewidth = 0.5) +
  scale_fill_manual(
    name = "Prioridade",
    values = c(
      "Extremamente Alta" = "green4",
      "Muito Alta" = "turquoise2",
      "Alta" = "blue2"
    ),
    na.value = "gray90"
  ) +
  theme_minimal() +
  labs(
    title = "Priorização dos corredores (remoção): Lethocerinae",
    fill = "dPC"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )
p1


# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(links_makurhini$importancia_total, n = 5, style =
"quantile")

# Crear una nueva variable categórica con los intervalos
links_makurhini <- links_makurhini %>%
  mutate(import_q = cut(importancia_total,
                        breaks = unique(breaks$brks),
                        include.lowest = TRUE,
                        dig.lab = 5))

# Criar mapa
p1 <- ggplot() +
  # Fundo (estado/região)
  geom_sf(data = rioSul_reprojetado, fill = NA, color = "black",
          size = 0.5) +
  geom_sf(data = links_makurhini, aes(color = import_q),
          size = 0.2, alpha = 0.9, linewidth = 0.3) +
  scale_color_brewer(palette = "RdYlBu", direction = -1, name = "dImport (Q)") +
  # Camada 2: Áreas prioritárias (categórica)
  geom_sf(data = areas_pampaRS, aes(fill = Prior_acao),
          color = 'white', linewidth = 0.5) +
  scale_fill_manual(
    name = "Prioridade",
    values = c(
      "Extremamente Alta" = "green4",
      "Muito Alta" = "turquoise2",
      "Alta" = "blue2"
    ),
    na.value = "gray90"
  ) +
  theme_minimal() +
  labs(
    title = "Importância total (remove): cenário 2",
    fill = "dImport"
```

```
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )
p1

# ---------- Top links ----------
# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(top_links_cen2$dPC_removal, n = 5, style =
"quantile")

# Crear una nueva variable categórica con los intervalos
top_links_cen2 <- top_links_cen2 %>%
  mutate(dPC_q = cut(dPC_removal,
                     breaks = unique(breaks$brks),
                     include.lowest = TRUE,
                     dig.lab = 5))

# Criar mapa
p1 <- ggplot() +
  # Fundo (estado/região)
  geom_sf(data = rioSul_reprojetado, fill = NA, color = "black",
          size = 0.5) +
  geom_sf(data = top_links_cen2, aes(color = dPC_q),
          size = 0.2, alpha = 0.9, linewidth = 0.3) +
  scale_color_brewer(palette = "RdYlBu", direction = -1, name = "dPC remove (Q)") +
  # Camada 2: Áreas prioritárias (categórica)
  geom_sf(data = areas_pampaRS, aes(fill = Prior_acao),
          color = 'white', linewidth = 0.5) +
  scale_fill_manual(
    name = "Prioridade",
    values = c(
      "Extremamente Alta" = "green4",
      "Muito Alta" = "turquoise2",
      "Alta" = "blue2"
    ),
    na.value = "gray90"
  ) +
  theme_minimal() +
  labs(
    title = "Priorización de enlaces (remove): cenário 2",
    fill = "dPC"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )
p1


# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(top_links_cen2$importancia_total, n = 5, style =
"quantile")

# Crear una nueva variable categórica con los intervalos
top_links_cen2 <- top_links_cen2 %>%
  mutate(import_q = cut(importancia_total,
                        breaks = unique(breaks$brks),
                        include.lowest = TRUE,
                        dig.lab = 5))

# Criar mapa
p1 <- ggplot() +
  # Fundo (estado/região)
  geom_sf(data = rioSul_reprojetado, fill = NA, color = "black",
          size = 0.5) +
  geom_sf(data = top_links_cen2, aes(color = import_q),
          size = 0.2, alpha = 0.9, linewidth = 0.3) +
  scale_color_brewer(palette = "RdYlBu", direction = -1, name = "dImport (Q)") +
  # Camada 2: Áreas prioritárias (categórica)
  geom_sf(data = areas_pampaRS, aes(fill = Prior_acao),
          color = 'white', linewidth = 0.5) +
  scale_fill_manual(
    name = "Prioridade",
    values = c(
      "Extremamente Alta" = "green4",
```

```
      "Muito Alta" = "turquoise2",
      "Alta" = "blue2"
    ),
    na.value = "gray90"
  ) +
  theme_minimal() +
  labs(
    title = "Importância total: q95 (remove) - cenário 2",
    fill = "dImport"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )
p1


# ----------------------- CENÁRIO 3: LETHOCERINAE + BELOSTOMATINAE ------------------
library(gdistance)  # Para cálculo de corredores
library(scales)     # Para normalização

areas_prioritarias = areas_pampaRS
superficie_resistencia = resist3
nome_cenario = 'TOTAL'

resistance_df <- as.data.frame(superficie_resistencia, xy = TRUE)
head(resistance_df)
dim(resistance_df)
colnames(resistance_df) <- c("x", "y", "value")

library(ggnewscale)

# Código corrigido:
p <- ggplot() +
  # Camada 1: Resistência (contínua)
  geom_tile(data = resistance_df, aes(x = x, y = y, fill = value), alpha = 0.8) +
  scale_fill_gradientn(
    name = "Resistência",
    colors = c("#000004FF", "#1B0C42FF", "#4B0C6BFF", "#781C6DFF",
               "#A52C60FF", "#CF4446FF", "#ED6925FF", "#FB9A06FF",
               "#F7D03CFF", "#FCFFA4FF")
  ) +

  # NOVA ESCALA DE FILL (reset)
  new_scale_fill() +

  # Camada 2: Áreas prioritárias (categórica)
  geom_sf(data = areas_pampaRS, aes(fill = Prior_acao),
          color = 'white', linewidth = 0.5) +
  scale_fill_manual(
    name = "Prioridade",
    values = c(
      "Extremamente Alta" = "green4",
      "Muito Alta" = "turquoise2",
      "Alta" = "blue2"
    ),
    na.value = "gray90"
  ) +

  # Camada 3: Contorno do estado
  geom_sf(data = rioSul_reprojetado, fill = NA, color = "black",
          linewidth = 0.4) +

  # Tema
  theme_minimal() +
  theme(
    axis.title.x = element_blank(),
    axis.title.y = element_blank(),
    legend.position = "right"
  )

print(p)

library(purrr)
# A los valores NA les asignamos un alto valor para evitar que pasen por ahí
superficie_resistencia[is.na(superficie_resistencia)] <- 1000
```

```
superficie_raster <- raster(superficie_resistencia)

# Criar objeto de transição
tr <- transition(superficie_raster, transitionFunction =
                   function(x) 1/mean(x), directions = 8)
tr <- geoCorrection(tr, type = "c", multpl = FALSE)

# Extrair centroides das áreas prioritárias
centroides <- st_centroid(areas_prioritarias, of_largest_polygon = T)
coords <- st_coordinates(centroides)

# Calcular matriz de custo mínimo
# matriz_custo <- costDistance(tr, coords)

# resistencia_Total <- project(resistencia_Total, crs_projetado_metros)

#Loop para estimar corredores entre parches
rutas_list <- list()
counter <- 1
for (i in 1:(nrow(coords) - 1)) {
  #cat(paste0(i, " de ", nrow(centroides), "\r"))
  counter <- 1
  rutas <- map_dfr((i + 1):nrow(coords), function(j){
    if(counter <= nrow(coords)){
      ruta <- shortestPath(tr, coords[i,], coords[j,], output = "SpatialLines")
      ruta <- st_as_sf(ruta); st_crs(ruta) <- crs_projetado_metros
      ruta$from <-i ; ruta$to <- j
      return(ruta)
    }
  })
  rutas_list[[i]] <- rutas
}
rutas_mc_total <- do.call(rbind, rutas_list)
rutas_mc_total

ggplot() +
  geom_sf(data = rioSul_reprojetado, fill = NA, color = "black") +
  geom_sf(data = rutas_mc_total, aes(color = "corredores"), color = "black",
linewidth = 0.5) +
  # Camada 2: Áreas prioritárias (categórica)
  geom_sf(data = areas_pampaRS, aes(fill = Prior_acao),
          color = 'white', linewidth = 0.3) +
  scale_fill_manual(
    name = "Prioridade",
    values = c(
      "Extremamente Alta" = "green4",
      "Muito Alta" = "turquoise2",
      "Alta" = "blue2"
    ),
    na.value = "gray90"
  ) +
  theme_minimal() +
  labs(
    title = "Corredores potenciales"
  ) +
  theme(
    plot.title = element_text(hjust = 0.5)
  )

#Distancia efectiva promedio como umbral de distancia
# Considerar proporção de espécies em cada grupo
n_belostomatinae <- dim(stack_adequabilidades_Belostoma)[3]  # Número de espécies
Belostomatinae
n_lethocerinae <- dim(stack_adequabilidades_Lethocerus)[3]     # Número de espécies
Lethocerinae
total_especies <- n_belostomatinae + n_lethocerinae

# Peso proporcional
peso_belo <- n_belostomatinae / total_especies
peso_letho <- n_lethocerinae / total_especies

# Distância combinada ponderada
Effec_mean_combinado <- mean(superficie_raster[], na.rm = TRUE) * 82000 *
  (peso_belo * 1.0 + peso_letho * 2.3)
Effec_mean_combinado # Resultado: 11985995

# Effec_mean <- mean(superficie_raster[], na.rm = TRUE) * 82000 # 82km
```

```
#Aplicamos la función
delta_cen3 <- MK_dPCIIC_links(nodes = areas_pampaRS,
                              attribute = 'riqueza_media',
                              area_unit = "ha",
                              distance = list(type = "least-cost",
                                              resistance = superficie_raster,
                                              least_cost.java = F,
                                              cores.java = 1),
                              removal = TRUE,
                              metric = "PC",
                              probability = 0.5,
                              distance_thresholds = round(Effec_mean_combinado),
                              # parallel = 6,
                              # parallel_mode = 0,
                              intern = TRUE,
                              write =
paste0("D:/Dropbox/curso_conectividade/resultados_projeto/", ecorreg,
                                           "/dPCIIC_links_LethoBelos_results"))

head(delta_cen3)

cat("✓ Links calculados com sucesso:", nrow(delta_cen3), "links\n")

#Existen otras formas, pero crearé un nuevo ID
delta_cen3$ID_nuevo <- paste0(delta_cen3$Destination, "_", delta_cen3$Source)

#Guardo las rutas en un objeto nuevo para tener de respaldo mi vector original
rutas_mc2_lethoBelos <- rutas_mc_total
rutas_mc2_lethoBelos$ID_nuevo <- paste0(rutas_mc2_lethoBelos$from, "_",
rutas_mc_total$to)

#Aplicar merge
rutas_mc2_lethoBelos <- merge(rutas_mc2_lethoBelos, delta_cen3, by = "ID_nuevo")

# Adicionar informações do cenário
rutas_mc2_lethoBelos$cenario <- nome_cenario
rutas_mc2_lethoBelos$distance_threshold <- round(Effec_mean_combinado)

# Mostrar resumo dos links
cat("Resumo dos links:\n")
cat("- dPC médio:", round(mean(rutas_mc2_lethoBelos$dPC_removal, na.rm = TRUE), 6),
"\n")
cat("- dPC máximo:", round(max(rutas_mc2_lethoBelos$dPC_removal, na.rm = TRUE), 6),
"\n")
cat("- dPC mínimo:", round(min(rutas_mc2_lethoBelos$dPC, na.rm = TRUE), 6), "\n")

links_makurhini = rutas_mc2_lethoBelos

# ============================================================================
# CRITÉRIO 1: CUSTO DO LINK (baseado em dPC)
# Maior dPC = menor custo = maior importância
# ============================================================================

custo_min <- min(links_makurhini$dPC_removal, na.rm = TRUE)
custo_max <- max(links_makurhini$dPC_removal, na.rm = TRUE)

if(custo_max > custo_min) {
  links_makurhini$importancia_custo <- (links_makurhini$dPC_removal - custo_min) /
(custo_max - custo_min)
} else {
  links_makurhini$importancia_custo <- 0.5  # Valor neutro se todos iguais
}

# ============================================================================
# CRITÉRIO 2: RIQUEZA DAS ÁREAS CONECTADAS
# ============================================================================
# Extrair riqueza das áreas conectadas
riqueza_origem <- areas_prioritarias$riqueza_media[match(links_makurhini$Source,
1:nrow(areas_prioritarias))]
riqueza_destino <-
areas_prioritarias$riqueza_media[match(links_makurhini$Destination,
1:nrow(areas_prioritarias))]

# Tratar NAs (caso alguma área não seja encontrada)
riqueza_origem[is.na(riqueza_origem)] <- 0
riqueza_destino[is.na(riqueza_destino)] <- 0
```

```
# Calcular riqueza média da conexão
links_makurhini$riqueza_origem <- riqueza_origem
links_makurhini$riqueza_destino <- riqueza_destino
links_makurhini$riqueza_media_conexao <- (riqueza_origem + riqueza_destino) / 2

# Normalizar riqueza
riqueza_min <- min(links_makurhini$riqueza_media_conexao, na.rm = TRUE)
riqueza_max <- max(links_makurhini$riqueza_media_conexao, na.rm = TRUE)

if(riqueza_max > riqueza_min) {
  links_makurhini$importancia_riqueza <- (links_makurhini$riqueza_media_conexao -
riqueza_min) / (riqueza_max - riqueza_min)
} else {
  links_makurhini$importancia_riqueza <- 0.5
}

# ============================================================================
# CRITÉRIO 3: ÁREA DAS PATCHES CONECTADAS
# ============================================================================
# Extrair áreas das patches
area_origem <- areas_prioritarias$Area_ha[match(links_makurhini$Source,
1:nrow(areas_prioritarias))]
area_destino <- areas_prioritarias$Area_ha[match(links_makurhini$Destination,
1:nrow(areas_prioritarias))]

# Tratar NAs
area_origem[is.na(area_origem)] <- 0
area_destino[is.na(area_destino)] <- 0

# Calcular área média da conexão
links_makurhini$area_origem <- area_origem
links_makurhini$area_destino <- area_destino
links_makurhini$area_media_conexao <- (area_origem + area_destino) / 2

# Normalizar área
area_min <- min(links_makurhini$area_media_conexao, na.rm = TRUE)
area_max <- max(links_makurhini$area_media_conexao, na.rm = TRUE)

if(area_max > area_min) {
  links_makurhini$importancia_area <- (links_makurhini$area_media_conexao - area_min)
/ (area_max - area_min)
} else {
  links_makurhini$importancia_area <- 0.5
}

# ============================================================================
# IMPORTÂNCIA CUMULATIVA (média dos 3 critérios)
# ============================================================================
# Média aritmética dos 3 critérios
links_makurhini$importancia_total <- (links_makurhini$importancia_custo +
                                      links_makurhini$importancia_riqueza +
                                      links_makurhini$importancia_area) / 3

# Classificar por importância (maior = mais importante)
links_makurhini <- links_makurhini[order(links_makurhini$importancia_total,
decreasing = TRUE), ]
links_makurhini$rank_importancia <- 1:nrow(links_makurhini)

# Criar categorias de importância
q95 <- quantile(links_makurhini$importancia_total, 0.95, na.rm = TRUE)
q75 <- quantile(links_makurhini$importancia_total, 0.75, na.rm = TRUE)
q50 <- quantile(links_makurhini$importancia_total, 0.50, na.rm = TRUE)
q25 <- quantile(links_makurhini$importancia_total, 0.25, na.rm = TRUE)

links_makurhini$categoria_importancia <- case_when(
  links_makurhini$importancia_total >= q95 ~ "Extremamente Alta",
  links_makurhini$importancia_total >= q75 ~ "Muito Alta",
  links_makurhini$importancia_total >= q50 ~ "Alta",
  links_makurhini$importancia_total >= q25 ~ "Moderada",
  TRUE ~ "Baixa"
)
print(table(links_makurhini$categoria_importancia))

# Salvar resultados
```

```
st_write(links_makurhini,
"D:/Dropbox/curso_conectividade/resultados_projeto/PAMPA/corredores_importantes_cenar
io3.shp",
         delete_dsn = TRUE)

# Salvar tabela CSV
tabela_corredores_cen3 <- st_drop_geometry(links_makurhini)
write.csv(tabela_corredores_cen3,

"D:/Dropbox/curso_conectividade/resultados_projeto/PAMPA/corredores_importantes_cenar
io3.csv", row.names = FALSE)

if(nrow(links_makurhini) > 0) {
  cat("Link mais importante:\n")
  cat("- Origem:", links_makurhini$from[1], "→ Destino:", links_makurhini$to[1],
"\n")
  cat("- Importância total:", round(links_makurhini$importancia_total[1], 3), "\n")
  cat("- dPC:", round(links_makurhini$dPC_removal[1], 6), "\n")
  cat("- Riqueza média:", round(links_makurhini$riqueza_media_conexao[1], 2), "\n")
  cat("- Área média:", round(links_makurhini$area_media_conexao[1], 2), "ha\n")
}

# Salvar apenas os mais importantes (top 25%)
top_links_cen3 <- links_makurhini[links_makurhini$categoria_importancia %in%
c("Extremamente Alta"), ]
write.csv(top_links_cen3,
"D:/Dropbox/curso_conectividade/resultados_projeto/PAMPA/top_links_cenario3.csv",
row.names = FALSE)

# resultados_finais <- list()
resultados_finais$cenario3 <- links_makurhini

subtitulo <- paste("Links de importância -",
                   nrow(top_links_cen3), "de", nrow(links_makurhini), "links")

head(top_links_cen3)

# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(links_makurhini$dPC_removal, n = 5, style =
"quantile")

# Crear una nueva variable categórica con los intervalos
links_makurhini <- links_makurhini %>%
  mutate(dPC_q = cut(dPC_removal,
                     breaks = unique(breaks$brks),
                     include.lowest = TRUE,
                     dig.lab = 5))

# Criar mapa
p1 <- ggplot() +
  # Fundo (estado/região)
  geom_sf(data = rioSul_reprojetado, fill = NA, color = "black",
          size = 0.5) +
  geom_sf(data = links_makurhini, aes(color = dPC_q),
          size = 0.2, alpha = 0.9, linewidth = 0.3) +
  scale_color_brewer(palette = "RdYlBu", direction = -1, name = "dPC remove (Q)") +
  # Camada 2: Áreas prioritárias (categórica)
  geom_sf(data = areas_pampaRS, aes(fill = Prior_acao),
          color = 'white', linewidth = 0.5) +
  scale_fill_manual(
    name = "Prioridade",
    values = c(
      "Extremamente Alta" = "green4",
      "Muito Alta" = "turquoise2",
      "Alta" = "blue2"
    ),
    na.value = "gray90"
  ) +
  theme_minimal() +
  labs(
    title = "Priorização dos corredores (remoção): TOTAL",
    fill = "dPC"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )
```

```
p1


# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(links_makurhini$importancia_total, n = 5, style =
"quantile")

# Crear una nueva variable categórica con los intervalos
links_makurhini <- links_makurhini %>%
  mutate(import_q = cut(importancia_total,
                        breaks = unique(breaks$brks),
                        include.lowest = TRUE,
                        dig.lab = 5))

# Criar mapa
p1 <- ggplot() +
  # Fundo (estado/região)
  geom_sf(data = rioSul_reprojetado, fill = NA, color = "black",
          size = 0.5) +
  geom_sf(data = links_makurhini, aes(color = import_q),
          size = 0.2, alpha = 0.9, linewidth = 0.3) +
  scale_color_brewer(palette = "RdYlBu", direction = -1, name = "dImport (Q)") +
  # Camada 2: Áreas prioritárias (categórica)
  geom_sf(data = areas_pampaRS, aes(fill = Prior_acao),
          color = 'white', linewidth = 0.5) +
  scale_fill_manual(
    name = "Prioridade",
    values = c(
      "Extremamente Alta" = "green4",
      "Muito Alta" = "turquoise2",
      "Alta" = "blue2"
    ),
    na.value = "gray90"
  ) +
  theme_minimal() +
  labs(
    title = "Importância total (remove): TOTAL",
    fill = "dImport"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )
p1

# ---------- Top links ----------
# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(top_links_cen3$dPC_removal, n = 5, style =
"quantile")

# Crear una nueva variable categórica con los intervalos
top_links_cen3 <- top_links_cen3 %>%
  mutate(dPC_q = cut(dPC_removal,
                     breaks = unique(breaks$brks),
                     include.lowest = TRUE,
                     dig.lab = 5))

# Criar mapa
p1 <- ggplot() +
  # Fundo (estado/região)
  geom_sf(data = rioSul_reprojetado, fill = NA, color = "black",
          size = 0.5) +
  geom_sf(data = top_links_cen3, aes(color = dPC_q),
          size = 0.2, alpha = 0.9, linewidth = 0.3) +
  scale_color_brewer(palette = "RdYlBu", direction = -1, name = "dPC remove (Q)") +
  # Camada 2: Áreas prioritárias (categórica)
  geom_sf(data = areas_pampaRS, aes(fill = Prior_acao),
          color = 'white', linewidth = 0.5) +
  scale_fill_manual(
    name = "Prioridade",
    values = c(
      "Extremamente Alta" = "green4",
      "Muito Alta" = "turquoise2",
      "Alta" = "blue2"
    ),
    na.value = "gray90"
  ) +
```

```
  theme_minimal() +
  labs(
    title = "Priorización de enlaces (remove): TOTAL",
    fill = "dPC"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )
p1


# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(top_links_cen3$importancia_total, n = 5, style =
"quantile")

# Crear una nueva variable categórica con los intervalos
top_links_cen3 <- top_links_cen3 %>%
  mutate(import_q = cut(importancia_total,
                        breaks = unique(breaks$brks),
                        include.lowest = TRUE,
                        dig.lab = 5))

# Criar mapa
p1 <- ggplot() +
  # Fundo (estado/região)
  geom_sf(data = rioSul_reprojetado, fill = NA, color = "black",
          size = 0.5) +
  geom_sf(data = top_links_cen3, aes(color = import_q),
          size = 0.5, alpha = 0.9, linewidth = 0.5) +
  scale_color_brewer(palette = "RdYlBu", direction = -1, name = "dImport (Q)") +
  # Camada 2: Áreas prioritárias (categórica)
  geom_sf(data = areas_pampaRS, aes(fill = Prior_acao),
          color = 'white', linewidth = 0.5) +
  scale_fill_manual(
    name = "Prioridade",
    values = c(
      "Extremamente Alta" = "green4",
      "Muito Alta" = "turquoise2",
      "Alta" = "blue2"
    ),
    na.value = "gray90"
  ) +
  theme_minimal() +
  labs(
    title = "Importância total: q95 (remove) - TOTAL",
    fill = "dImport"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )
p1
```